(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
10 May 2002 (10.05.2002)

PCT

(10) International Publication Number
WO 02/37859 A2

(51) International Patent Classification[7]: H04N 7/24

(21) International Application Number: PCT/US01/47222

(22) International Filing Date:
2 November 2001 (02.11.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
60/245,748 3 November 2000 (03.11.2000) US

(63) Related by continuation (CON) or continuation-in-part (CIP) to earlier application:
US 60/245,748 (CIP)
Filed on 3 November 2000 (03.11.2000)

(71) Applicant *(for all designated States except US)*: **COMPRESSION SCIENCE** [US/US]; 901 Campisi Way, Campbell, CA 95008 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **HAMILTON, Eric** [US/US]; 15690 Gum Tree Lane, Los Gatos, CA 95032 (US). **DOUGLAS, John** [US/US]; 1104 King Street, Santa Cruz, CA 95060 (US). **LELESCU, Dan** [CA/US]; 4400 The Woods Drive, Apt. 503, San Jose, CA 95116 (US). **FU, Dongshan** [CN/US]; 2250 Monroe Street #174, Santa Clara, CA 95050 (US). **SHI, Fang** [CA/US]; 225 Catalpa Avenue, Suite 307, San Mateo, CA 94401 (US). **WIDERGREN, Robert** [US/US]; 15054A Downing Oak Ct., Los Gatos, CA 95032 (US). **TESCHER, Andrew, G.** [US/US]; 14670 Fieldstone Drive, Saratoga, CA 95070 (US).

(74) Agent: **WEITZ, David, J.**; Wilson Sonsini Goodrich & Rosati, 650 Page Mill Road, Palo Alto, CA 94304-1050 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 02/37859 A2

(54) Title: VIDEO DATA COMPRESSION SYSTEM

(57) Abstract: Methods, software and systems are provided for encoding video data into a more compressed format and for decoding video data from the compressed format. Using these methods, software and systems, video data is preprocessed, encoded in a compressed format using local and global motion estimation, transmitted, decoded, and then postprocessed.

VIDEO DATA COMPRESSION SYSTEM

BACKGROUND OF THE INVENTION

Field of the Invention

This invention relates to software, methods and systems for video and audio data compression. More specifically, the invention relates to software, methods and systems for providing enhanced compression of video and audio data over that which is achieved by existing MPEG-2 technology.

Description of Related Art

Developments and improvements in electronic communications systems have enhanced the manner in which information may be transmitted. In particular, the capabilities of real-time video and audio systems have greatly improved in recent years. Nevertheless, transmission of real-time video and audio data continues to require a relatively-large bandwidth. In order to provide services such as video-on-demand, videoconferencing to subscribers, or multimedia wireless communications, a significant amount of bandwidth may be required. Bandwidth is often a main inhibitor of the effectiveness of such systems.

To reduce the constraints imposed by the limited bandwidth available in telecommunications networks, compression systems and standards have evolved. Certain such standards define formats for compression of video and audio data and for transmission of multiple programs and control data streams in a single bitstream. Various standards for compression and encoding of multimedia data, including video and audio, have been proposed and developed, with different degrees of market acceptance.

The Moving Picture Experts Group ("MPEG") standard constitutes one such video and audio compression standard. The MPEG standard was defined and is administered by the International Standards

Organization. MPEG-2, a particular adaptation of the MPEG standard, is described in the International Standards Organization -- Moving Picture Experts Group, Drafts of Recommendation H.262, ISO/IEC 13818 (various parts) titled "Information Technology -- Generic Coding Of Moving Pictures and Associated Audio" (hereinafter "the November 1993 ISO-MPEG Committee draft"). References made herein to MPEG-2 refer to MPEG-2 standards as defined in the November 1993 ISO-MPEG Committee drafts.

Although compression standards such as MPEG-2 have alleviated constraints associated with efficient encoding or transmission of video and audio data to a certain extent, such standards fail to perform adequately in certain operational environments. For example, storage media currently in wide commercial use, such as compact discs, can only accommodate a limited amount of MPEG-2 data, which may not be able to encode sufficiently long video sequences. A compact disc read-only memory ("CD-ROM"), a particular type of compact disc that is currently in wide commercial use, can store approximately 650 megabytes of data, which may be insufficient to store a full feature MPEG-2 movie. An emerging optical disk technology, digital versatile disk ("DVD"), is expected to replace CD-ROM technology in the future. Although a single-layer, single-side DVD may hold almost 5 gigabytes of information, this storage capacity can only currently store approximately two hours of MPEG-2 data. Consequently, MPEG-2 movies that are longer than approximately two hours may require more than one disk for storage, which may be undesirable or impractical. Furthermore, movies stored in a higher resolution format than what is provided by MPEG-2 require more storage capacity.

Another operational environment in which data compression and transmission exhibits performance limitations is satellite broadcasting. Satellites orbiting the Earth at high altitudes can conduct line-of-sight communications with terminals located throughout broad geographical areas and consequently provide an efficient mechanism for broadcasting

to a large number of receivers. A particular application of satellite communications is television broadcasting to ground-based subscriber receivers. Such receivers may be either stationary (e.g., antennas affixed to ground-based structures) or mobile (e.g., airplane on-board television systems).

Communications in satellite systems often experience bandwidth constraints due to various factors. Since the frequency spectrum allocated to satellite communications is administratively regulated and constrained, and since a large number of satellite systems compete for fractions of this frequency spectrum, the bandwidth available to any single satellite system is limited. Consequently, in satellite television broadcasting systems, the limited bandwidth available constrains the number of television channels that may be accommodated.

Figure 1A illustrates the transmission of compressed video from a video source to a plurality of terminals through satellite retransmission. A satellite system 100 comprises a satellite 102, an uplink transmitter 104 and receivers 108. The uplink transmitter 104 comprises an encoder 106 and provides line-of-sight uplink transmissions to the satellite 102. Receivers 108 comprise a plurality of stationary or mobile terminals, which receive data transmitted by the satellite 102. In operation, the encoder 106 encodes video information, possibly as MPEG-2 data or some other compressed format, and transmits the encoded information to the satellite 102 via the uplink transmitter 104. The satellite 102 receives the encoded information and rebroadcasts it to the terminals 108.

As previously discussed, the frequency spectrum available to any single satellite system is limited. For the satellite system 100, this may translate into bandwidth constraints for both the propagation path between the uplink transmitter 104 and the satellite 102 and for the wireless links between the satellite 102 and the terminals 108. Consequently, the number of television channels that may be broadcast to the terminals 108 is constrained. Since the frequency spectrum

available to the satellite system 100 is generally fixed, one way to increase the television broadcasting capacity of the satellite system 100 is to compress the television signals. Although encoding of television signals as MPEG-2 data generally increases the number of television channels available to terrestrial receivers, satellite broadcasting systems typically exhaust the frequency spectra allocated to them and subscriber demand for additional television channels and information may not be met. The bandwidths constraints of satellite systems are likely to be exacerbated by current and future developments in interactive services and high definition television technology which tend to require even higher bandwidths.

Another arena in which bandwidth constraints are strained by current data compression and transmission is terrestrial wireless communications. Wide-scale penetration of wireless technology in the commercial consumer market has led to development and deployment of extensive wireless communications networks which support both voice and data communications. Wireless networks have been traditionally limited in the bandwidth and data transmission capabilities offered to subscribers. Recent migration from analog to digital data formats has done little to alleviate these limitations.

Generally, the frequency spectrum limitations in wireless communications are acute for a number of reasons, including the need to accommodate large numbers of users simultaneously. Various mechanisms for addressing these limitations have been developed, including frequency reuse in a geographical cellular structure and data compression. While reducing the size of the geographical cells increases the ability to reuse frequencies, thereby increasing the effective data transmission capacity of wireless networks, logistical and technological limitations mandate minimum sizes for the cells. Consequently, enhanced compression of data constitutes an effective and attractive approach to further increasing the capacity of wireless networks.

Figure 1B illustrates the transmission of compressed video from a video source over a wireless network. A wireless system 110 comprises a wireless transmitter 112 and a plurality of wireless devices 114. The wireless transmitter 112 communicates with each of the wireless devices 114. In a wireless communications network, the wireless devices 114 may be located in a single cell, or may be distributed throughout different cells. Communications between the wireless transmitter 112 and wireless devices 114 are often bandwidth constrained to such an extent that transmission of multimedia data may be impractical or impossible. Encoding of video and audio information with greater compression is needed to increase the ability of the wireless transmitter 112 to transmit such information to the wireless devices 114.

While bandwidth constraints may be acute in satellite and wireless communications systems, as discussed above, bandwidth constraints also exist in wired networks. Conventional dial-up telephone modem connections between client computers and servers also exhibit significant bandwidth constraints, which make transmission of multimedia data difficult or impractical. Data connections over television cable modems or employing Digital Subscriber Lines ("DSL") improve the ability to transmit and receive video and audio data encoded, but more efficient compression is still needed.

Figure 1C illustrates the transmission of compressed video from a video source over a network. A communications system 120 comprises a video source 121, a network 122 and a client devices 124. The video source 121 is connected to the network 122 via a wired connection, a satellite link, or a wireless channel. Client devices 124 are connected to the network 122 through various types of connections, including DSL, cable modem, telephone modem and wireless connections. As previously discussed, each of these connections may experience a limited bandwidth, which may constrain the ability of the video source 121 to transmit information to the client devices 124. Even if the network 122 can accommodate high data rates within its boundaries, the

bandwidth constraints of the connections between the network 122 and the video source 121 and between the network 122 and the client devices 124 nevertheless impair transmission of video and audio data to the client devices 124. Encoding of video and audio data as MPEG-2 data often fails to remedy these problems.

A need thus exists for compression of video and audio data having greater performance than the MPEG-2 standard. Methods and systems for compression of video and audio data are provided herein which contribute to a new compression system with greater performance.

SUMMARY OF THE INVENTION

The present invention provides various computer executed methods that may be, independently or in combination, incorporated into a system for compressing video and audio data.

Several different methods, computer executable logic, and systems are provided in regard to the use of decision logic for encoding different frame types.

In one embodiment, computer readable medium is provided which comprises: data encoding a sequence of frames of video in a compressed format, the encoded video frames comprising intra frames which do not rely on another frame to encode an image for that frame, predicted frames which rely on a preceding intra frame or a preceding predicted frame to encode an image for that frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein a periodicity of intra frames and/or predicted frames varies within the encoded sequence of video frames; and logic for decompressing the data into a less compressed video format.

According to this embodiment, the periodicity of intra frames, predicted frames, and bi-directional predicted frames may each independently vary within the encoded sequence of video frames.

In another embodiment, computer readable medium is provided which comprises: data encoding a sequence of frames of video in a compressed format, the encoded video frames comprising intra frames which do not rely on another frame to encode an image for that frame, predicted frames which rely on a preceding intra frame or a preceding predicted frame to encode an image for that frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein a periodicity of intra frames and/or predicted frames varies within the encoded sequence of video frames based on a relative efficiency of encoding a given frame as different frame types; and logic for decompressing the data into a less compressed video format.

According to this embodiment, the periodicity of intra frames and/or predicted frames and/or bi-directionally predicted frames optionally varies within the encoded sequence of video frames based on a combination of the relative efficiency of encoding a given frame as different frame types and an image quality cost for encoding the given frame as the different frame types.

In another embodiment, computer readable medium is provided which comprises: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding intra frames in the encoded sequence of video frames

with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as an intra frame or another frame type.

According to this embodiment, the periodicity of intra frames within the encoded sequence of video frames encoded by the logic optionally varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as an intra frame or another frame type.

Also according to this embodiment, the logic optionally analyzes whether it is more efficient to encode some of the frames as an intra frame and/or a predicted frame and/or a bi-directionally predicted frame.

Also according to this embodiment, the logic optionally uses more than one preceding frame to evaluate the efficiency of encoding a given frame as an intra frame. These preceding frames may be selected from the group of intra frames and predicted frames.

Also according to this embodiment, the logic optionally employs a whole frame or a still image encoder to encode intra frames. Alternatively, the logic may employ a JPEG 2000 encoder or a wavelet-based encoder to encode intra frames.

Also according to this embodiment, the logic may evaluate whether it is more efficient to encode a given frame as an intra frame by fully encoding or only partially encoding the frame as both an intra frame and another frame type such as a predicted frame and/or a bi-directional predicted frame.

Also according to this embodiment, the logic may optionally encodes video frames that have not already been encoded or may encode video frames that have been pre-encoded in an MPEG format.

In another embodiment, computer readable medium is provided which comprises: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that

rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding predicted frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as a predicted frame or another frame type.

According to this embodiment, the periodicity of predicted frames within the encoded sequence of video frames encoded by the logic optionally varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as a predicted frame or another frame type.

In another embodiment, computer readable medium is provided which comprises: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding bi-directional predicted frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as a bi-directional frame or another frame type.

According to this embodiment, the periodicity of bi-directional predicted frames within the encoded sequence of video frames encoded by the logic optionally varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as a bi-directional predicted frame or another frame type.

In another embodiment, an encoder for encoding video in a compressed format is provided, the encoder comprising: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding intra frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as an intra frame or another frame type.

According to this embodiment, the periodicity of intra frames within the encoded sequence of video frames encoded by the logic optionally varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as an intra frame or another frame type.

Also according to this embodiment, the logic may optionally use more than one preceding frame to evaluate the efficiency of encoding a given frame as an intra frame. The more than one preceding frame may be intra frames and/or predicted frames.

Also according to this embodiment, the logic optionally employs a whole frame or a still image encoder to encode intra frames. Alternatively, the logic may employ a JPEG 2000 encoder or a wavelet-based encoder to encode intra frames.

Also according to this embodiment, the logic may evaluate whether it is more efficient to encode a given frame as an intra frame by fully encoding or only partially encoding the frame as both an intra frame and another frame type such as a predicted frame and/or a bi-directional predicted frame.

Also according to this embodiment, the logic may optionally encode video frames that have not already been encoded or may encode video frames that have been pre-encoded in an MPEG format.

In another embodiment, an encoder is provided for encoding video in a compressed format, the encoder comprising: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding predicted frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as a predicted frame or another frame type.

According to this embodiment, the periodicity of predicted frames within the encoded sequence of video frames encoded by the logic optionally varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as a predicted frame or another frame type.

In another embodiment, an encoder is provided for encoding video in a compressed format, the encoder comprising: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding bi-directional predicted frames in the encoded

sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as a bi-directional frame or another frame type.

According to this embodiment, the periodicity of bi-directional predicted frames within the encoded sequence of video frames encoded by the logic optionally varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as a bi-directional predicted frame or another frame type.

In another embodiment, computer readable medium is provided which comprises: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein the logic comprises decision logic which evaluates whether to encode a given frame as an intra frame or another frame type.

According to this embodiment, the decision logic optionally evaluates whether to encode a given frame as an intra frame or another frame type based at least in part on whether it is more coding efficient to encode the given frame as an intra frame or another frame type. Examples of other types of frames include predicted frames, bi-directional predicted frames and super bi-directional predicted frames.

According to this embodiment, the decision logic optionally evaluates whether to encode a given frame as an intra frame or another frame type based at least in part on a combination of whether it is more coding efficient to encode the given frame as an intra frame or another frame type coding efficiency and an image quality cost function.

According to this embodiment, the decision logic optionally uses more than one preceding frame to evaluate whether to encode a given frame as an intra frame or another frame type. These preceding frames are optionally intra frames and/or predicted frames.

According to this embodiment, the decision logic optionally evaluates whether to encode a given frame as an intra frame or another frame type by fully encoding the given frame as both an intra frame and another frame type.

Alternatively, the decision logic may optionally evaluate whether to encode a given frame as an intra frame or another frame type by only partially encoding the given frame as both an intra frame and another frame type.

According to this embodiment, the logic for encoding video in a compressed format may be adapted to encode uncompressed video or may encode video that is already in a compressed format such as an MPEG format.

In another embodiment, an encoder is provided for encoding video in a compressed format, the encoder comprising: logic for receiving a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein the logic comprises decision logic which evaluates whether to encode a given frame as an intra frame or another frame type.

According to this embodiment, the decision logic optionally evaluates whether to encode a given frame as an intra frame or another frame type based at least in part on whether it is more coding efficient to encode the given frame as an intra frame or another frame type.

Also according to this embodiment, the decision logic optionally evaluates whether to encode a given frame as an intra frame or another frame type based at least in part on a combination of whether it is more coding efficient to encode the given frame as an intra frame or another frame type coding efficiency and an image quality cost function.

Also according to this embodiment, the decision logic optionally comprises logic for evaluating whether to encode a given frame as an intra frame or another frame type such as a predicted frame, a bi-directional predicted frame and/or a super bi-directional predicted frame.

Also according to this embodiment, the decision logic optionally uses more than one preceding frame to evaluate whether to encode a given frame as an intra frame or another frame type. The preceding frames may optionally be intra frames and/or predicted frames.

Also according to this embodiment, the decision logic optionally evaluates whether to encode a given frame as an intra frame or another frame type by fully encoding or only partially encoding the given frame as both an intra frame and another frame type.

Also according to this embodiment, the logic for encoding video in a compressed format is optionally adapted to encode video frames pre-encoded in an MPEG format.

In another embodiment, a computer executed method is provided for encoding a sequence of frames of video in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the computer executed method comprising: analyzing whether encoding a given frame of the sequence as an intra frame or another frame type is more coding efficient; using results from the analysis to decide whether to encode the given frame as an intra frame or another frame type; and encoding the given frame as

an intra frame or another frame type based on the decision. The another frame type may be a predicted frame, bi-directional predicted frame and/or a super bi-directional predicted frame.

According to this embodiment, the method may further comprise analyzing an image quality cost associated with encoding the given frame as an intra frame or another frame type; and using results from the analysis to decide whether to encode the given frame as an intra frame or another frame type comprises both results from the analysis of whether it is more coding efficient to encode the given frame as an intra frame or another frame type and the image quality cost results.

Also according to this embodiment, the analysis regarding whether encoding a given frame as an intra frame or another frame type is more coding efficient is optionally performed using more than one preceding frame selected from the group of intra frames and predicted frames as potential reference frames for the given frame.

Also according to this embodiment, the analysis may be performed by fully encoding the given frame or by only partially encoding the given frame as both an intra frame and at least one other frame type. The another frame type may be a predicted frame, bi-directional predicted frame and/or a super bi-directional predicted frame.

Also according to this embodiment, the sequence of frames of video to be encoded may be uncompressed or may already be in a compressed format such as an MPEG format.

Several different methods, computer executable logic, and systems are also provided in regard to the encoding and decoding super bi-directional predicted frames.

For example, in one embodiment, a method is provided for encoding a sequence of frames of video comprising: encoding a first group of frames as intra frames which do not rely on another frame to encode an image for that frame; encoding a second group of frames as predicted frames which rely on a preceding intra frame or a preceding predicted frame to encode an image for that frame; encoding a third

group of frames as bi-directional predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame; and encoding a fourth group of frames as super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame, and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, wherein at least a portion of the super bi-directional predicted frames are encoded with reference to at least one bi-directional frame.

According to this embodiment, at least a portion of the super bi-directional predicted frames may rely on a preceding bi-directional frame. Also according to this embodiment, at least a portion of the super bi-directional predicted frames may rely on a subsequent bi-directional frame.

Also according to this embodiment, at least a portion of the super bi-directional predicted frames may rely upon a preceding intra frame, predicted frame, or bi-directional frame which is not a frame immediately preceding the super bi-directional predicted frame.

Also according to this embodiment, at least a portion of the super bi-directional predicted frames may rely upon a subsequent intra frame, predicted frame, or bi-directional frame which is not a frame immediately following the super bi-directional predicted frame.

In another embodiment, computer readable medium is provided which comprises: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame

and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame.

According to this embodiment, the computer readable medium may further comprise logic for determining an available bandwidth for transmitting video in a compressed format and logic for transmitting video with or without super bi-directional predicted frames based on the determined available bandwidth.

In another embodiment, computer readable medium is provided comprising logic for decoding video from a compressed format, the logic comprising: logic for decoding intra frames that do not rely on another frame to encode an image for that frame; logic for decoding predicted frames that rely on a preceding intra frame or predicted frame, logic for decoding bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame; and logic for decoding super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame.

In another embodiment, a method is provided for decoding video from a compressed format, the method comprising: decoding a group of intra frames which do not rely on another frame to encode an image for that frame; decoding a group of predicted frames that rely on a preceding intra frame or predicted frame; decoding a group of bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame; and decoding a group of super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame, and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame.

Several different methods, computer executable logic, and systems are also provided in regard to the use of JPEG 2000 to encode intraframes. For example, in one embodiment, a computer readable medium is provided which comprises: logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding intra frames using a JPEG 2000 encoder.

Several different methods, computer executable logic, and systems are also provided in regard to scene change detection.

In one embodiment, computer readable medium is provided which encodes logic for detecting scene changes in a sequence of frames of video, the logic comprising: logic for encoding video in a compressed format by taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or preceding frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein the logic evaluates whether to encode a given frame as an intra frame or a predicted frame based at least in part on a relative coding efficiency between encoding the frame as an intra frame or a predicted frame; and logic which identifies a frame as being a beginning of a scene change based, at least in part, on the frame being encoded as an intraframe.

According to this embodiment, the decision logic may optionally evaluate whether to encode a given frame as an intra frame or a predicted frame based at least in part on a combination of whether it is

more coding efficient to encode the given frame as an intra frame or a predicted frame and an image quality cost function.

Also according to this embodiment, the decision logic may optionally evaluate whether to encode a given frame as an intra frame or a predicted frame by fully encoding the given frame as both an intra frame and a predicted frame.

According to this embodiment, the decision logic may also optionally evaluate whether to encode a given frame as an intra frame or a predicted frame by only partially encoding the given frame as both an intra frame and a predicted frame.

Several different methods, computer executable logic, and systems are also provided in regard to performing rate control.

In one embodiment, a method is provided for transmitting video in a compressed format, the compressed video comprising a sequence of frames that comprise intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, the method comprising: determining an amount of bandwidth required to transmit the sequence of frames with or without use of super bi-directionally predicted frames; determining an amount of bandwidth available at a given time to transmit the sequence of frames with or without use of super bi-directionally predicted frames; and transmitting the video data with or without use of super bi-directionally predicted frames based at least in part on whether the bandwidth available at the given time is sufficient to transmit the sequence of frames without the use of super bi-directionally predicted frames.

According to this embodiment, the super bi-directionally predicted frames used may optionally also be based in part on an image quality cost function associated with using the super bi-directionally predicted frames.

According to this embodiment, determining the amount of bandwidth required and determining the amount of bandwidth available may be performed continuously as the video data is transmitted, the video data being transmitted both with and without super bi-directionally predicted frames over time based on whether the bandwidth available at the given time is sufficient to transmit the sequence of frames without the use of super bi-directionally predicted frames.

Also according to this embodiment, determining the amount of bandwidth required and determining the amount of bandwidth available may be performed periodically as the video data is transmitted, the video data being transmitted both with and without super bi-directionally predicted frames over time based on whether the bandwidth available at the given time is sufficient to transmit the sequence of frames without the use of super bi-directionally predicted frames.

In another embodiment, computer readable medium is provided for use in a method for transmitting video in a compressed format, the compressed video comprising a sequence of frames that comprise intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, the method comprising: logic for determining an amount of bandwidth required to transmit the sequence of frames with or without use of super bi-directionally predicted frames; logic for determining an amount of

bandwidth available at a given time to transmit the sequence of frames with or without use of super bi-directionally predicted frames; and logic for causing the video data to be transmitted with or without use of super bi-directionally predicted frames based at least in part on whether the bandwidth available at the given time is sufficient to transmit the sequence of frames without the use of super bi-directionally predicted frames.

In another embodiment, a method is provided for decoding video transmitted in a compressed format, the compressed video comprising a sequence of frames that comprise intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, the method comprising: determining whether super bi-directionally predicted frames are being transmitted; and if super bi-directionally predicted frames are being transmitted, decoding the super bi-directionally predicted frames, or if super bi-directionally predicted frames are not being transmitted, decoding transmitted intra frames, predicted frames, and bi-directionally predicted frames, and producing additional frames based on the decoded transmitted intra frames, predicted frames, and bi-directionally predicted frames such that the resulting frame sequence has a desired frame rate.

In yet another embodiment, computer executable logic is provided for decoding video transmitted in a compressed format, the compressed video comprising a sequence of frames that comprise intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or

predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, the computer executable logic comprising: logic for determining whether super bi-directionally predicted frames are being transmitted; logic which, if super bi-directionally predicted frames are being transmitted, decodes the super bi-directionally predicted frames; and logic which, if super bi-directionally predicted frames are not being transmitted, decodes transmitted intra frames, predicted frames, and bi-directionally predicted frames, and produces additional frames based on the decoded transmitted intra frames, predicted frames, and bi-directionally predicted frames such that the resulting frame sequence has a desired frame rate.

Several different methods, computer executable logic, and systems are also provided in regard to performing local motion estimation.

In one embodiment, a hierarchical computer executed method is provided for use in a video compression system for determining motion vectors associated with a block for use in coding decisions, the method comprising: selecting a first block; selecting a second block which is a down-sampled block of the first block; determining multiple motion vectors for the second block; and refining motion vectors for the first block based, at least in part, on the motion vectors of the second block to produce a set of refined motion vectors for the first block to use in the encoding process.

According to this embodiment, refining the motion vectors may optionally be performed by a fractional pixel refinement process.

Also according to this embodiment, the second block may optionally have a size selected from the group consisting of 2x2, 4x4, and 8x8 and the first block has a size selected from the group consisting of 4x4, 8x8 and 16x16.

In another embodiment, a hierarchical computer executed method is provided for use in a video compression system for determining motion vectors associated with a block for use in coding decisions, the method comprising: selecting a first block; selecting a second block which is a down-sampled block of the first block; selecting a third block which is a down-sampled block of the second block; determining multiple motion vectors for the third block; refining motion vectors for the second block based, at least in part, on the motion vectors of the third block to produce a set of refined motion vectors for the second block; and refining motion vectors for the first block based, at least in part, on the motion vectors of the second block to produce a set of refined motion vectors for the first block to use in the encoding process.

According to one variation of this embodiment, the third block may has a size selected from the group consisting of 2x2, 4x4, and 8x8, the second block has a size selected from the group consisting of 4x4, 8x8 and the first block has a size selected from the group consisting of 8x8 and 16x16.

In another embodiment, a hierarchical computer executed method is provided for use in a video compression system for determining motion vectors associated with a block for use in coding decisions, the method comprising: selecting a first block; selecting a second block which is a down-sampled block of the first block; selecting a third block which is a down-sampled block of the second block; selecting a fourth block which is a down-sampled block of the third block; determining multiple motion vectors for the fourth block; refining motion vectors for the third block based, at least in part, on the motion vectors of the fourth block to produce a set of refined motion vectors for the third block; refining motion vectors for the second block based, at least in part, on the motion vectors of the third block to produce a set of refined motion vectors for the second block; and refining motion vectors for the first block based, at least in part, on the motion vectors of the second block

to produce a set of refined motion vectors for the first block to use in the encoding process.

According to one variation of this embodiment, the fourth block has a 2x2 size, the third block has a 4x4 size, the second block has a 8x8 size, and the first block has a 16x16 size.

According to any of these embodiments for determining motion vectors, the multiple motion vectors may comprise at least 2, 3, 4 or more candidate motion vectors per block.

Also according to any of these embodiments for determining motion vectors, refining the motion vectors may optionally be performed by a fractional pixel refinement process.

Several different methods, computer executable logic, and systems are also provided for performing multi-pass motion vector refinement by applying SAD criteria to neighboring block motion vectors.

In one embodiment, a method is provided for refining motion vectors for blocks for a given frame, the method comprising: a) taking a motion vector for a given block and a set of motion vectors for blocks neighboring the given block; b) computing SAD criteria for the given block using the motion vector for the given block and the set of motion vectors for the neighboring blocks; c) selecting which of the motion vector for the given block and the set of motion vectors for the neighboring blocks has the smallest SAD criteria for the given block; and d) if the motion vector with the smallest SAD criteria is not the motion vector for the given block, replacing the motion vector for the given block with the motion vector with the smallest SAD; wherein steps a) – d) are repeated for all blocks in the given frame until no block's motion vector is replaced by a neighboring block's motion vector.

In another embodiment, a method is provided for refining motion vectors for blocks for a given frame, the method comprising: a) taking a motion vector for a given block and a set of motion vectors for blocks neighboring the given block; b) computing SAD criteria for the given

block using the motion vector for the given block and the set of motion vectors for the neighboring blocks; c) selecting which of the motion vector for the given block and the set of motion vectors for the neighboring blocks has the smallest SAD criteria for the given block; and d) if the motion vector with the smallest SAD criteria is not the motion vector for the given block, replacing the motion vector for the given block with the motion vector with the smallest SAD; wherein steps a) – d) are repeated for multiple iterations.

According to either of these embodiments, the given block may be a block or macroblock. For example, the given block may have a size selected from the group consisting of 4x4, 4x8, 8x4, 8x8, 16x8, 8x16, and 16x16.

Several different methods, computer executable logic, and systems are also provided for performing motion vector refinement using larger blocks to refine motion vectors of smaller blocks.

In one embodiment, a local motion estimation method is provided which comprises: estimating one or more motion vectors for a macroblock; subdividing the macroblock into a first set of blocks where each block in the set is smaller than the macroblock; using the one or more motion vectors for a macroblock to estimate one or more motion vectors for each of the blocks in the first set; subdividing each of the blocks in the first set into second sets of blocks where each block in the second sets of blocks are smaller than the blocks in the first set of blocks; and using the one or more motion vectors for each of the blocks in the first set to estimate one or more motion vectors for each of the blocks in the second set of blocks.

According to this embodiment, at least 2, 3, 4 or more motion vectors may be estimated for each block or macroblock.

According to this embodiment, the first set of blocks which the macroblock is subdivided into may optionally have the same size. Optionally, the first set of blocks which the macroblock are subdivided into rectangular or square blocks.

According to this embodiment, estimating the motion vectors may optionally be performed by searching around pixel matrices centered upon one of the motion vectors for a block from which the block was subdivided.

Also according to this embodiment, estimating the motion vectors may also optionally be performed by searching using a SAD cost function.

Several different methods, computer executable logic, and systems are also provided for performing motion vector refinement to a high degree of accuracy.

In another embodiment, a method is provided for performing motion estimation with non-integer pixel precision, the method comprising: a) taking a first set of motion vectors mapping a given block to a first reference block; b) determining a second set of motion vectors by up-sampling the first set of motion vectors and the reference frame to determine a second set of motion vectors which map to a second reference block in the upsampled reference frame; and c) repeating steps a) and b) where the motion vectors of step b) are employed as the motion vectors taken in step a) until an $1/8^{th}$ pixel precision level is reached, after which, the resulting determined motion vectors are employed in an encoding process.

In regard to this embodiment, determining the second set of motion vectors may optionally be performed by performing a local search around the upsampled first set of motion vectors.

Also in regard to this embodiment, the first set of motion vectors may optionally comprise at least 2, 3, 4 or more candidate motion vectors per block.

Several different methods, computer executable logic and systems are also provided which use predictors for motion vectors based on previously encoded motion vectors.

Several different methods, computer executable logic and systems are also provided for predicting local motion vectors for a frame being encoded.

In one embodiment, a method is provided for predicting local motion vectors for a frame being encoded, the method comprising: a) taking a motion vector for a given block of a frame to be encoded which maps a motion of the given block relative to a reference frame; b) taking a set of candidate motion vectors which map motion of blocks neighboring the given block relative to one or more reference frames; c) identifying members of the set of candidate motion vectors which are not validated based on one or more validation rules; d) compensating for any identified non-validated candidate motion vectors; e) scaling the validated and compensated motion vectors with respect to the frame being encoded; and f) computing a predictor motion vector for the motion vector for the given block based on the scaled motion vectors.

According to this embodiment, the method may further comprise encoding an error determined by comparing the predictor motion vector and the motion vector for the given block.

Also according to this embodiment, the candidate motion vectors may comprise motion vectors for macroblocks in the frame being encoded and/or blocks within the macroblocks.

Also according to this embodiment, four of the candidate motion vectors for the given block may be selected based on a position of the given block within a macroblock within the frame being encoded.

Also according to this embodiment, one of the candidate motion vectors in the set may be a motion vector used by a block in a reference frame.

Also according to this embodiment, four of the candidate motion vectors for the given block may be selected based on a position of the given block within a macroblock within the frame being encoded, and a fifth of the candidate motion vectors in the set may be a motion vector used by a block in a reference frame.

Also according to this embodiment, one of the candidate motion vectors in the set may be a motion vector used by a block in a reference frame that is in a same position in the reference frame as the given block in the frame being encoded.

Also according to this embodiment, four of the candidate motion vectors for the given block may be selected based on a position of the given block within a macroblock within the frame being encoded, and a fifth of the candidate motion vectors in the set is a motion vector used by a block in a reference frame that is in a same position in the reference frame as the given block in the frame being encoded.

Also according to this embodiment, at least one candidate motion vector may optionally be identified as being invalid.

Also according to this embodiment, scaling of motion vectors may optionally be performed by the formula

$$ScaledMVi = ((t0-t1)/(t1-t2)) * MVi$$

where

Mvi is a candidate motion vector predictor,

ScaledMVi is the scaled candidate motion vector predictor,

t0 is time of current motion vector,

t1 is the time of the frame to which the current motion vector references, and

t2 is the time of the frame, to which a motion vector in the co-located block of the frame at time t1, references.

Also according to this embodiment, one or more of the validation rules used in the method may be selected from the group consisting of: if any candidate motion vector is not coded, it is invalid; if only one candidate motion vector is invalid, it is set to zero; if two candidate motion vectors are invalid, the two motion vectors are discarded; if three candidate motion vectors are invalid, two are discarded and a third is set to zero; if four candidate motion vectors are invalid, they are each set to

a fifth motion vector candidate; and if five candidate motion vectors are invalid, they are each set to zero.

In another embodiment, a method is provided for predicting local motion vectors for a frame being encoded, the method comprising: a) taking a motion vector for a given block of a frame to be encoded which maps a motion of the given block relative to a reference frame; b) taking candidate motion vectors which map motion of blocks neighboring the given block relative to one or more reference frames; c) identifying members of the candidate motion vectors which are not validated based on one or more validation rules; d) compensating for any identified non-validated candidate motion vectors; e) scaling the validated and compensated motion vectors with respect to the frame being encoded; f) identifying an additional candidate motion vector by taking an average of the scaled motion vectors and identifying a motion vector for a block in a reference frame to which the average motion vector points; g) identifying whether the additional candidate motion vector is valid based on one or more validation rules and compensating for the additional candidate motion vector if it is invalid; h) scaling the additional candidate motion vector with respect to the frame being encoded; and i) computing a predictor motion vector for the motion vector for the given block based on the scaled candidate motion vectors.

Several different methods, computer executable logic and systems are also provided for computing affine transform coefficients for global motion estimation using local motion segmentation.

In one embodiment, a method is provided for estimating an affine model, the method comprising: a) taking an initial set of motion vectors which map macroblocks in a frame to be encoded to corresponding macroblocks in a reference frame; b) determining a set of affine equations based on the initial set of motion vectors; c) determining an affine model by solving the determined set of affine equations; d) determining a modified set of motion vectors based on the determined affine model; e) eliminating motion vectors from the modified set of

motion vectors which are inconsistent with the initial motion vector; and f) determining a final affine model by repeating steps b-e where the motion vectors determined in step d which are not eliminated in step e are used as the initial set of motion vectors in step b until either (i) a predetermined number of iterations have been performed, or (ii) a predetermined accuracy threshold for the modified motion vectors has been reached.

According to this embodiment, the set of affine equations may optionally comprise at least six equations. The set of affine equations may also optionally exceed a number of affine model parameters used to determine the set of affine equations. Optionally, initial values for the affine parameters are determined using a least squares estimation.

In one variation, the affine model is expressed by the equation

$$u_k = a_1 x_k + b_1 y_k + c_1, \text{ and}$$

$$v_k = a_2 x_k + b_2 y_k + c_2$$

where u_k and v_k are predicted x and y components of a motion vector corresponding to a macroblock k in the frame being encoded and variables a_1, b_1, c_1, a_2, b_2 and c_2 are affine model parameters to be determined. According to this variation, values for u_k and v_k may be determined based on motion vectors derived during local motion estimation.

In another variation, determining the modified set of motion vectors may be based on the determined set of affine equations is performed by using the affine model to construct a predicted frame on a pixel-by-pixel basis.

In yet another variation, eliminating motion vectors from the modified set of motion vectors which are inconsistent with the affine model may be performed by selecting a filtering threshold defining a maximum allowed deviation from a corresponding motion vector derived during local motion estimation, and eliminating those motion vectors which do not satisfy the threshold.

In another variation, steps a-d may be repeated for at least 2, 3, 4, 5, 6, 7, 8, 9 or more iterations. Optionally, steps a-d are repeated until the set of motion vectors determined in step c and not eliminated in step d satisfy a final accuracy threshold.

The accuracy thresholds employed in the multiple iterations of steps a-d may optionally be pre-defined for each iteration. In one variation, the accuracy threshold comprises a magnitude threshold and a phase threshold. The accuracy threshold preferably decreases with each iteration. In one particular variation, the final accuracy threshold is 0.5 pixels for a magnitude of the motion vectors and 5 degrees for the phase of the motion vectors.

Several different methods, computer executable logic and systems are also provided for encoding bidirectionally predicted frames using global motion estimation.

In one embodiment, a method is provided for encoding a bidirectionally predicted frame, the method comprising: computing an affine model for a given bi-directionally predicted frame to be encoded where a preceding frame is used as a reference frame in the affine model; warping the preceding reference frame; determining a residue between the given bi-directionally predicted frame to be encoded and the warped preceding reference frame; and encoding the given bi-directionally predicted frame to be encoded with reference to the residue.

In another embodiment, a method is provided for encoding a bidirectionally predicted frame, the method comprising: computing an affine model for a given bi-directionally predicted frame to be encoded where a subsequent frame is used as a reference frame in the affine model; warping the subsequent reference frame; determining a residue between the given bi-directionally predicted frame to be encoded and the warped subsequent reference frame; encoding the given bi-directionally predicted frame to be encoded with reference to the residue.

According to each of these embodiments, the affine model is optionally computed by a) taking an initial set of motion vectors which map macroblocks in a frame to be encoded to corresponding macroblocks in a reference frame; b) determining a set of affine equations based on the initial set of motion vectors; c) determining an affine model by solving the determined set of affine equations; d) determining a modified set of motion vectors based on the determined affine model; e) eliminating motion vectors from the modified set of motion vectors which are inconsistent with the initial motion vector; and f) determining a final affine model by repeating steps b-e where the motion vectors determined in step d which are not eliminated in step e are used as the initial set of motion vectors in step b until either (i) a predetermined number of iterations have been performed, or (ii) a predetermined accuracy threshold for the modified motion vectors has been reached.

Several different methods, computer executable logic and systems are also provided which use local motion vector(s) to refine block predictions based on global motion estimation.

In one embodiment, a method is provided for refining a global motion estimation for a given block of a current frame being encoded, the method comprising: computing an affine model for the current frame; warping a reference frame for the current frame; determining a prediction error between the given block in the current frame and a block within the warped reference frame; determining motion vectors between the given block and the block within the warped reference frame; and modifying the prediction error based on the determined motion vectors.

According to this embodiment, the affine model is optionally computed by a) taking an initial set of motion vectors which map macroblocks in a frame to be encoded to corresponding macroblocks in a reference frame; b) determining a set of affine equations based on the initial set of motion vectors; c) determining an affine model by solving the determined set of affine equations; d) determining a modified set of

motion vectors based on the determined affine model; e) eliminating motion vectors from the modified set of motion vectors which are inconsistent with the initial motion vector; and f) determining a final affine model by repeating steps b-e where the motion vectors determined in step d which are not eliminated in step e are used as the initial set of motion vectors in step b until either (i) a predetermined number of iterations have been performed, or (ii) a predetermined accuracy threshold for the modified motion vectors has been reached.

Several different methods, computer executable logic and systems are also provided which use multiple hypothesis in combination with multiple reference frames to encode blocks.

In one embodiment, a method is provided for encoding a given block employing multiple hypotheses, the method comprising: taking multiple reference blocks from multiple reference frames; and encoding the given block based on a combination of the multiple reference blocks.

In one variation, taking multiple reference macroblocks comprises selecting the multiple reference blocks from a larger set of reference blocks by taking groups of reference blocks of the larger set of reference macroblocks and selecting a subset of those groups based on a cost function. Optionally, the cost function for each group of reference blocks may be based on a cost of encoding the given macroblock with respect to a combination of reference blocks. Optionally, two or more of the groups of reference blocks comprise a same block.

In another variation, the combination of the multiple reference blocks may be combined to comprise a predictor block for the block being encoded.

In another variation, the groups of reference blocks comprise at least two, three, four or more blocks.

It is noted that the given block and reference block may be a block or a macroblock. Each optionally has a size selected from the group consisting of 4x4, 4x8, 8x4, 8x8, 16x8, 8x16, and 16x16.

Several different methods, computer executable logic and systems are also provided which use overlapped block motion compensation with multiple reference frames for block encoding.

In one embodiment, a method is provided for encoding a given block of a macroblock, the method comprising: encoding the given block on a pixel by pixel basis using a predictor pixel value obtained by combining reference pixel values from multiple reference frames as indicated by motion vectors for two blocks neighboring the given block and a motion vector for the given block.

In one variation, the motion vectors for two blocks neighboring the given blocks have been modified by translation of their coordinates from their original coordinates to coordinates of a current pixel being encoded.

Optionally, combining reference pixel values may comprise weighing the reference pixel values from the multiple reference frames according to the position of the current pixel being encoded in the given block.

Also according to the embodiment, the pixel being encoded in the given block may be encoded with respect to the predictor value obtained by weighing the reference pixel values from the multiple reference frames according to the position of the current pixel.

Several different methods, computer executable logic and systems are also provided which use a minimum rate-distortion cost function to make coding decisions.

In one embodiment, a method is provided for deciding a coding mode for a given block based on a minimum rate-distortion cost function, the method comprising: taking multiple reference blocks from multiple reference frames; taking multiple coding modes for the given block with respect to a reference block or a combination of reference blocks; and determining a coding decision for the given block by minimizing a cost as determined by a rate-distortion cost function.

According to this embodiment, determining the coding decision may comprise selecting a reference block or a combination of reference blocks for a given block, and selecting a block coding mode with respect to the reference block or a combination of reference blocks based on the rate-distortion cost function. Selecting the block coding mode for the given block with respect to a reference block or a combination of reference blocks may optionally comprise selecting the coding mode from a set of all possible block coding modes based on the rate-distortion cost function.

Also according to this embodiment, the rate-distortion cost function employs a weighted combination of a distortion cost function and a rate cost function for a given block with respect to a selected reference block or a combination of reference blocks and the selected block coding mode. Optionally , the rate-distortion cost function may be determined by: taking transformed coefficients of the current block; taking a scan order for grouping the transformed coefficients of the current block; grouping one or more transformed coefficient to form a set of coefficients; modifying attributes of each of the coefficients in the set based on a rate-distortion cost function for the set of the coefficients; repeating the steps of grouping and modifying iteratively until all of the transformed coefficients are processed; calculating the distortion cost function for the given block based on the modified transformed coefficients of the given block; and calculating the rate cost function for the given block based on the modified transformed coefficients of the given block. Optionally, the rate-distortion cost function for the set of the coefficients comprises a weighted combination of the distortion cost function and the rate cost function of the given set of transformed coefficients

Also according to this embodiment, the combination of reference blocks may comprise at least two blocks. The given block and reference block may be blocks or macroblocks and may have a size selected from the group consisting of 4x4, 4x8, 8x4, 8x8, 16x8, 8x16, and 16x16.

Several different methods, computer executable logic and systems are also provided which use energy compaction when performing block encoding.

In one embodiment, a method is provided for encoding a given macroblock, the method comprising: a) taking a current macroblock of a current frame; b) determining a reference macroblock of a different, reference frame for the current macroblock based on motion estimation; c) selecting for a block from the current macroblock a corresponding block from the reference macroblock based on a block level motion estimation for the current block or based on a corresponding position in the macroblock; d) sorting pixels for each line of the reference block based on pixel values within the line; e) identifying a permutation for each line in the sorted reference block corresponding to a modified order of the pixels in the line as a result of the sorting; f) permuting pixels for each line of the current block based on the corresponding permutation for each line of the corresponding block of the reference macroblock; and g) calculating a prediction error block based on a difference between the permuted current block and the sorted reference block; wherein steps a – g are repeated for all blocks of the current macroblock and a frequency transformation is applied to each of the blocks of the residual macroblock so obtained.

In another embodiment, a method is provided for decoding a macroblock, the method comprising: receiving a prediction error block in a bitstream which corresponds to a current block to be decoded; taking the corresponding reference block from an already decoded reference frame for the current block to be decoded; sorting pixels for each line of the reference block based on pixel values within the line; identifying a permutation for each line in the sorted reference block corresponding to a modified order of the pixels in the line as a result of the sorting; adding the sorted reference block to the corresponding prediction error block to obtain a permuted current block; and using the identified line

permutations to inverse permute the permuted current block lines to obtain the reconstructed current block.

In another embodiment, a method is provided for encoding a given macroblock, the method comprising: a) taking a prediction error block from a prediction error macroblock for the given macroblock being encoded; b) for each line of the prediction error block: permute the line in all possible combinations to generate all possible permutations, optimally match the different permutations generated to a target signal from a target signal matrix which comprises targets signals used in transforming the prediction error block, identify which of the matched permutations has a lowest cost with respect to a target signal based on a cost function, permute all lines in the prediction error block according to the permutation for the current line identified as having the lowest cost, optimally match each line of the permuted prediction error block to a target signal from the target signal matrix based on a cost function, and determine and record a cumulative block level cost by summing the costs for each optimally matched line and associate the determined cumulative block level cost with the current line being processed from the prediction error block; c) determine a minimum block level cost from the cumulative block level costs determined in step b corresponding to each line from the prediction error block, and record the associated permutation; d) use the permutation identified in step c to permute all the lines in the prediction error block; and e) determine and transmit results of a frequency transform of the permuted prediction error block determined in step d.

In another embodiment, a method is provided for encoding a given macroblock, the method comprising: a) determining a block for the given macroblock being encoded; b) for each line of the block: permute the line in all possible combinations to generate all possible permutations, optimally match the different permutations generated to a target signal from a target signal matrix which comprises targets signals used in transforming the block, identify which of the matched

permutations has a lowest cost with respect to a target signal based on a cost function, permute all lines in the block according to the permutation for the current line identified as having the lowest cost, optimally match each line of the permuted block to a target signal from the target signal matrix based on a cost function, and determine and record a cumulative block level cost by summing the costs for each optimally matched line and associate the determined cumulative block level cost with the current line being processed from the block; c) determine a minimum block level cost from the cumulative block level costs determined in step b corresponding to each line from the block, and record the associated permutation; d) use the permutation identified in step c to permute all the lines in the block; and e) determine and transmit results of a frequency transform of the permuted block determined in step d.

In another embodiment, a method is provided for decoding a macroblock, the method comprising: receiving a permuted prediction error block from a bitstream which corresponds to a current block to be decoded; applying an inverse permutation which was used to encode the prediction error block to the prediction error block to produce an inverse permuted prediction error block; and adding the inverse permuted prediction error block to an already decoded reference block to obtain a reconstructed current block.

In yet another embodiment, a method is provided for decoding a macroblock comprising: receiving a permuted block from a bitstream; and applying an inverse permutation which was used to encode the block to the block to produce an inverse permuted block.

In yet another embodiment, a method is provided for encoding a given macroblock comprising: a) taking a current macroblock of a current frame; b) determining a reference macroblock of a different, reference frame for the current macroblock based on motion estimation; c) selecting for a block from the current macroblock a corresponding block from the reference macroblock based on macroblock level motion

estimation; d) sorting pixels for each line of the reference block based on pixel values within the sorted line; e) identifying a permutation for each line corresponding to a modified order of the pixels in the line as a result of the sorting; f) permuting pixels for each line of the current block based on the permutations for each line of the corresponding block of the reference macroblock.; and g) calculating a prediction error block based on a comparison between the permuted current block and the sorted reference block; wherein steps a – g are repeated for all blocks of the current macroblock and a frequency transformation is applied to each of the blocks of the residual macroblock so transformed.

It is noted in regard to the energy compaction embodiments, that the processes can operate on the rows or columns of the block being processed. In this regard, columns should be construed as a form of row, orthogonal to another form of row. It is also noted that energy compaction may also be performed in a sequence, by first processing a block row-by-row (or column-by-column), and then processing the result in a column-by-column (or row-by-row) fashion.

Several different methods, computer executable logic and systems are also provided for performing smoothing images.

In one embodiment, a method is provided for reducing block artifacts in decoded images, the method comprising: taking a decoded block having edge pixels defining a perimeter of the block; and for each edge pixel: taking an edge pixel as a current pixel, determining a local variance for the current pixel by comparing the current pixel to pixels neighboring the current pixel, determining absolute differences between two neighboring pixels of the current pixel a) normal to the edge, b) in a positive diagonal direction relative to the edge, and c) in a negative diagonal direction relative to the edge, constructing a pixel-based directional filter based on the determined local variance and the determined absolute differences, and calculating a new value for the current pixel using the constructed pixel-based directional filter.

In another embodiment, a method is provided for smoothing artifacts within a decoded image, the method comprising: constructing a cost function including a closeness term and a smoothness term; determining a system of linear equations by minimizing the cost function; and iteratively solving the equations for smoothed values for all pixels.

Several different methods, computer executable logic and systems are also provided for post processing decoded images.

In one embodiment, a method is provided for reducing contours in a decoded image, the method comprising: taking a decoded image; for blocks of the decoded image, computing local variances for each block; using the computed local variances to determine smooth regions of the decoded image; segmenting the detected smooth regions; randomly selecting pixels within each smooth region; and adding random noise to the selected pixels.

It is noted in regard to the above descriptions that although some embodiments are only described in terms of a method, computer executable logic and/or system, it should be understood that this is not intended to be limiting. Rather, incorporation of a method into computer executable logic and/or system would be well understood in the art and is certainly intended within the scope of the present invention.

It is also noted in regard to the above descriptions that these embodiments are only exemplary and that it should be recognized that further embodiments of the present invention are also described herein.

BRIEF DESCRIPTION OF THE FIGURES

Figure 1A illustrates the transmission of compressed video from a video source to a plurality of terminals through satellite retransmission.

Figure 1B illustrates the transmission of compressed video from a video source over a wireless network.

Figure 1C illustrates the transmission of compressed video from a video source over a network.

Figure 2A illustrates a series of frames compressed according to an embodiment of the MPEG 2 format.

Figure 2B illustrates a series of frames compressed according to an aspect of the present invention.

Figure 3A shows a flow chart illustrating a method for deciding whether to encode a particular frame as an I frame or a P frame, according to an aspect of the present invention.

Figure 3B shows a flow chart illustrating an alternative method for deciding whether to encode a particular frame as an I frame or a P frame, according to an aspect of the present invention.

Figure 3C shows an embodiment of a method for scene change detection based on a decision to encode a frame as an I frame or a P frame.

Figure 3D illustrates a series of frame sequences where scene changes may be detected according to the embodiment of Figure 3C.

Figure 3E illustrates a sequence of frames compressed according to the present invention where super B (SB) frames are employed.

Figure 4A illustrates a layer structure comprising an enhanced frame layer for improved data compression according to an embodiment of the present invention.

Figure 4B illustrates a layer structure comprising two enhanced frame layers for improved data compression according to an embodiment of the present invention.

Figure 5 illustrates a schematic diagram of a system for compressing video that comprises various aspects of the present invention.

Figure 6A illustrates a flow diagram describing an embodiment of motion estimation logic that may be used in conjunction with an aspect of the present invention.

Figure 6B illustrates a flow chart of a local motion estimation process that may be used in conjunction with motion estimation according to an aspect of the present invention.

Figure 6C illustrates a flow diagram for multiple pass motion vector refinement according to an aspect of the present invention.

Figure 6D illustrates a flow diagram for block-level local motion estimation according to an aspect of the present invention.

Figure 6E illustrates a flow diagram for fractional pixel motion vector refinement according to an aspect of the present invention.

Figure 6F illustrates an embodiment of global motion compensation that may be used in conjunction with motion estimation according to an aspect of the present invention.

Figure 6G illustrates a flow chart of an affine model estimation and motion segmentation process that may be used in connection with global motion estimation according to an aspect of the present invention.

Figure 7A illustrates a system for encoding a P frame based on selecting an encoding mode from a plurality of encoding modes according to an embodiment of the present invention.

Figure 7B illustrates a system for encoding a bi-directionally predicted frame based on selecting an encoding mode from a plurality of encoding modes according to an embodiment of the present invention.

Figure 7C illustrates a system for encoding a macroblock using a multiple hypothesis macroblock encoding mode according to an embodiment of the present invention.

Figure 7D illustrates a flow diagram for encoding a macroblock using an overlapped motion compensation macroblock encoding mode according to an embodiment of the present invention.

Figure 7E illustrates a context that may serve as a basis for encoding a current block based on an overlapped motion compensation encoding mode, according to an embodiment of the present invention.

Figure 8A illustrates a system for rate distortion optimization that may be employed to select a preferred macroblock encoding mode in accordance with an aspect of the present invention.

Figure 8B illustrates a method for indirect energy compaction based on a permutation of data, according to an aspect of the present invention.

Figure 8C illustrates a method for direct energy compaction with respect to an energy-normalized basis function based on a permutation of the underlying data, according to an aspect of the present invention.

Figure 8D illustrates an adaptive context that may serve as a basis for encoding data into a bitstream in a zig-zagged scanning pattern, according to an aspect of the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention provides various computer executed methods that may be, independently or in combination, incorporated into a system for compressing video and audio data. Such systems may be used to encode, decode or transcode data and may be incorporated into various devices for performing those functions. The methods provided in accordance with various embodiments of the present invention may be implemented in hardware, in software, or as a combination of hardware and software.

For example, dedicated hardware or computer executable software comprising various methods provided in accordance with aspects of the present invention may be deployed within the encoder 106 from Figure 1A to encode data transmitted to the satellite 102, within the satellite 102 to perform transcoding, and/or within the terminals 108 to decode the data received from the satellite 102. Similarly, various embodiments of the present invention may be incorporated as hardware and/or software within the wireless transmitter 112 from Figure 1B to enhance compression of the data transmitted to the wireless devices

114, and within the wireless devices 114 to decompress the data. Embodiments of the present invention may analogously be implemented as hardware and/or computer executable software within the video source 121 from Figure 1C to encode data transmitted via the network 122 to the client devices 124. Utilization of methods provided by various aspects of the invention in connection with such systems improves the level of data compression achieved by the resulting system while maintaining or improving the corresponding signal fidelity and image quality.

It is noted that the present invention is also intended to encompass compressed data, storage media comprising compressed data, and electronic wave forms encoding compressed data, where the compressed data is formed by the methods, software and/or systems of the present invention.

Through the advances in data compression achieved by the methods, software, and systems of the present invention, existing challenges in data transmission and storage are addressed. For example, higher visual and audio fidelity and richer multimedia content may be transmitted or received using the same or less bandwidth that is currently required by MPEG-2 data encoding methods.

1. FRAME TYPES AND FRAME TYPE ENCODING DECISIONS

Video is formed by the sequential display of multiple frames, typically at least 30 frames per second. The image associated with each frame is data-encoded. Since at least a portion of an image associated with a given frame (referred to herein as picture element) may appear in another temporally-proximate frame, data compression can be achieved by using a temporally-proximate frame as a reference to at least partially data-encode another frame. This process of video compression by

establishing references between frames is commonly referred to as "temporal compression."

Temporal compression relies on the basic assumption that consecutive frames in a video frame sequence are generally correlated and include a significant amount of redundant information. For an encoding rate of 30 frames per second, each video frame spans approximately 3 milliseconds. For typical video sequences, objects captured in consecutive video frames experience few changes in such a short time span. This inherent property is exploited in temporal compression by using immediately preceding and/or following frames to detect and express changes in the location and characteristics of the picture elements present in the respective frames while ignoring similarities. Temporal compression uses a process referred to as motion compensation, described in greater detail herein, in which a motion vector is used to describe the translation of picture elements from one frame to another.

In the context of temporal compression, frames may be classified according to their role in the compression process roles as Intra frames, Predicted frames, Bi-directionally predicted frames and Super bi-directionally predicted frames.

An Intra frame ("I" frame) is a frame that does not rely on another frame to encode the image for that frame. Because I frames do not reference other frames, they are typically data-costly to encode. However, because I frames do not reference any other frames, they can be independently decoded and therefore provide useful starting points for decoding video. This is particularly important in applications where video is being transmitted and decoding must begin somewhere after the beginning of the transmission. For example, when multiple channels are being broadcast, I frames provide decoding entry points for the different channels.

A predicted frame ("P" frame) is a frame that relies on a preceding I frame or P frame to encode the corresponding image. By

referencing the preceding I frame or P frame, some degree of data compression relative to encoding the frame as an I frame is frequently achieved. It is noted that the referenced preceding I or P frames need not be the first preceding I or P frame.

A Bi-directionally predicted frame ("B" frame) is a frame that relies on a preceding I or P frame and/or a subsequent I or P frame to encode the corresponding image. B frames, however, do not rely on other B frames to encode the image for that frame. It is noted that the referenced preceding I or P frames need not be the first preceding I or P frame. Similarly, the referenced subsequent I or P frames need not be the first subsequent I or P frame.

By referencing two temporally-proximate frames, a greater degree of data compression relative to encoding the frame as either an I or P frame is frequently achieved. Since a B frame may reference an I frame or P frame that follows it, decoding of B frames may require buffering of reference frames at the decoder to reconstruct the original frame structure.

A. Frame Implementation In MPEG-2

Figure 2A illustrates a series of frames compressed according to an embodiment of the MPEG 2 format. The particular frame distribution pattern illustrated in Figure 2A corresponds to a pattern widely adopted and commonly used in the industry. This pattern has been employed in encoding MPEG-2 data with a frame rate of 30 frames per second. In this implementation, frames 1-15 collectively span half of a second, and their distribution pattern repeats every half of a second.

In the distribution pattern of Figure 2A, an I frame is placed every 15 frames. Consequently, with a frame rate of 30 frames per second, two I frames are encoded every second (i.e., at positions 1 and 16 in the periodic frame sequence).

Figure 2A also shows a series of P frames, with one P frame encoded every three frames after the initial I frame (i.e., at positions 4, 7,

10 and 13). Each P frame refers back to the first I frame or P frame that immediately precedes the frame. Consequently, at 30 frames per second, a total of eight P frames are encoded every second.

The frame distribution pattern of Figure 2A also includes a set of B frames collocated in pairs and immediately following every I frame or P frame in the sequence (i.e., B frames are located at positions 2, 3, 5, 6, 8, 9, 11, 12, 14 and 15). Each B frame may refer back to the I frame or P frame that immediately precedes the frame and/or refer forward to the I frame or P frame that immediately follows the frame. At 30 frames per second, a total of twenty B frames are encoded every second.

As previously mentioned, I, P and B frames are used for temporal compression. Temporal compression exploits the fact that contiguous video frames contain a significant amount of redundant information by encoding certain frames in reference to other frames and attempts to eliminate the redundancies. Figure 2A illustrates an implementation of temporal compression commonly used in MPEG-2. By their inherent nature, each of the I frames shown in Figure 2A at positions 1 and 16 is fully encoded without reference to any other frames. The P frame at position 4, however, is encoded in reference to the I frame at position 1. Similarly, the P frame at position 7 is encoded in reference to the P frame at position 4. Each of the B frames at positions 2 and 3 may be encoded in reference to the I frame at position 1 and/or the P frame at position 4.

MPEG-2 implementations typically impose a fixed periodicity for I frames, regardless of the characteristics of the underlying video sequence. For example, in the frame sequence of Figure 2A, there are at least 2 I frames every 30 frames and this pattern does not change with the content of the video images. One reason for this may be that I frames, by their nature do not refer to any other preceding or subsequent frames. As a result, the frame can be decoded independent of other frames.

The ability to decode a given frame independently of other frames is attractive for addressing channel changes. When a channel is changed, decoding of the next channel cannot occur until an I frame that can be independently decoded is encountered. As illustrated in Figure 2A, an I frame is encoded every 15 frames and therefore will be encountered within a half second of a channel change at a frame rate of 30 frames per second.

B. Frame Type Encoding And Encoding Decisions

In certain instances where the rate of change of images in video is very rapid, it may prove to be more data-efficient to encode a particular frame independently (i.e., as an I frame) than with reference to another frame (i.e., as a P frame). In other instances, where the rate of change of images in video is not as rapid, it may prove to be more data-efficient to encode a give frame with reference to another frame (i.e., as a P or B frame), as opposed to encoding the frame independently (i.e., as an I frame). Since the rate of image change may vary frequently during the course of a video sequence, the relative efficiency between encoding a frame as an I frame or P frame may change many times, and at different time or frame intervals.

One aspect of the present invention relates to not fixing the periodicity for I frames and P frames. Instead, the characteristics of the underlying video sequence are analyzed to determine whether a given frame would be more efficiently encoded as an I frame or a P frame. This can be achieved by using an evaluation of coding efficiency to determine whether to code a frame as an I frame or a P frame. Frames are encoded as I frames when it is more efficient to do so and are encoded as P frames when it is more efficient to do so. Frames that may be encoded as either an I frame or a P frame according to this aspect of the invention are denoted as "I/P" frames. The ability of the present invention to vary the coding of frames as I or P frames based on a relative coding efficiency provides a further measure of coding

efficiency, thereby lowering the global data rate and/or providing higher signal fidelity at the same or lower global data rate.

Because it may sometimes be more efficient to encode a particular I/P frame as an I frame and it may sometimes be more efficient to encode a particular I/P frame as a P frame, the periodicity of I and P frames may each independently vary within a sequence of data compressed according to an embodiment of the present invention. For example, Figure 2B illustrates a series of frames compressed according to one aspect of the present invention. In Figure 2B, frames 4, 7, 10, 13, and 16 are illustrated as being either an I frame or a P frame. Accordingly, each of frames 4, 7, 10, 13, and 16 may independently be either I or P frames. Hence, frames 4, 7, 10, 13, and 16 can assume 32 different combinations of I and P frames.

It is noted in regard to Figure 2B that this aspect of the present invention does not require that there be a given number of frames between each I/P frame. Although the figure shows two B frames between each I/P frame, one, two, three, four, five, six, seven, eight, nine, or more frames may be positioned between each I/P frame. For example, more than two frames are preferably placed between each I/P frame when Super-B frames, to be described herein, are employed.

Figure 3A provides a flow chart illustrating a method that may be used in conjunction with the present invention to decide whether to encode a particular frame as an I frame or a P frame based on the format that provides more efficient data compression. It is noted that more efficient data compression may optionally be based on the number of bits required to encode the image or may be based on a combination of the number of bits required to encode the image and a cost function relating to the image quality of the resulting image. It is noted that other methods for determining whether a given frame is more efficiently encoded as an I frame or P frame may also be implemented.

In the embodiment illustrated in Figure 3A, the decision to encode a frame as an I frame or a P frame comprises encoding the frame in

both formats and selecting the format that provides maximum compression.

According to the method of Figure 3A, a frame is selected from the video sequence in step 302. The frame selected in step 302 is then encoded as an I frame in step 304, and the encoded I frame is stored in a retrievable manner. Either prior to, simultaneously with, or subsequent to steps 304 or 306, the frame selected in step 302 is encoded as a P frame in step 308, and the encoded P frame is stored in step 310.

The sizes of the memory spaces required to store the encoded I and P frames are compared in step 312 to determine the encoding format that provides the highest degree of compression. In one embodiment of the invention, the sizes of the memory spaces required to store the encoded I and P frames are determined by referring to the amount of memory occupied by the two encoded frames as stored in steps 306 and, respectively, 310. In an embodiment, the sizes of the memory spaces required to store the encoded I and P frames are determined and stored at the time when the frames are encoded in steps 304 and, respectively, 308, and are retrieved in step 312.

If the size of the memory space required to store the I frame is smaller than the size of the memory space required for the P frame, the frame selected in step 302 is classified as an I frame in step 314. The encoded I frame stored in step 306 is then retrieved in step 316 and is introduced in the frame sequence at the appropriate position. If the size of the memory space required to store the I frame is larger than the size of the memory space required for the P frame, the frame selected in step 302 is classified as a P frame in step 318. The encoded P frame stored in step 310 is then retrieved in step 320 and is introduced in the frame sequence at the appropriate position.

In alternative embodiments, the encoded I and P frames are not stored in steps 306 and, respectively, 310. In those alternative embodiments, retrieval of the encoded I and P frames in steps 316 or, respectively, 320 does not take place. Instead, depending on whether

the frame selected in step 302 was classified as an I frame or P frame in steps 314, or, respectively, 318, the frame is encoded again as an I frame or P frame prior to being inserted in the frame sequence.

It is noted that the process illustrated in Figure 3A for evaluating whether to encode a frame as an I or P frame may be performed for only a single preceding I/P frame. Alternatively, multiple preceding I/P frames may be used to evaluate whether to encode the frame as an I or P frame. The number of preceding I/P frames used to evaluate whether to encode the frame as an I or P frame may be fixed or varied.

Figure 3B shows a flow chart illustrating an alternative method for deciding whether to encode a particular frame as an I frame or a P frame. While the method of Figure 3A selects the encoding format of a frame based on full encoding of the frame in I and P formats, the method of Figure 3B selects the encoding format of a frame based on cost functions corresponding to the two encoding formats. Such cost functions may include various criteria that estimate compression efficiency and are further described below.

In the method shown in Figure 3B, the decision process commences by selecting a particular frame to be encoded. This step is similar to step 302 comprised in the method of Figure 3A.

In step 324, the method of Figure 3B then determines a cost function C_I for the frame selected in step 322 corresponding to an I frame encoding format. This cost function may comprise partial encoding as an I frame and/or determination of peak signal-to-noise ratio.

In step 326, a cost function C_P is determined, which may comprise motion vector estimation, partial encoding, and/or determination of peak signal-to-noise ratio.

As was noted in regard to Figure 3A, the process of Figure 3B for evaluating whether to encode a frame as an I or P frame may be performed for only a single preceding I/P frame. Alternatively, multiple preceding I/P frames may be used to evaluate whether to encode the

frame as an I or P frame. The number of preceding I/P frames used to evaluate whether to encode the frame as an I or P frame may be fixed or varied. When multiple preceding I/P frames are used to evaluate whether to encode the frame as an I or P frame, the process of selecting in step 322 is repeated for the multiple preceding I/P frames and cost functions are determined for the multiple preceding I/P frames.

The cost functions C_I and C_P determined in steps 324 and 326 are compared at step 330. In an embodiment, multiple cost functions associated with one or more I frames are also determined and are compared with one or more P frame cost functions. Comparison of cost functions in step 330 provides a basis for selecting a preferred frame encoding format.

In a particular embodiment, cost functions may include estimation of a number of bits required to encode the frame selected in step 322 as an I frame and, alternatively, as a P frame. To determine this number of bits, the underlying frame may be fully or partially encoded. The encoded bitstream corresponding to the frame provides a measure of the number of bits required to encode the frame. In this embodiment, the decision in step 330 may comprise selecting the format that results in the lowest number of encoded bits, or correspondingly, the highest compression ratio. The decision in step 330 may also comprise selecting the format that provides the highest peak signal to noise ratio.

If the comparison performed in step 330 determines that the cost function associated with the I frame format is superior to one or more of the cost functions associated with the P frame format, the frame selected in step 322 is classified as an I frame in step 332 and is encoded as an I frame in step 334. Otherwise, the frame selected in step 322 is classified as a P frame in step 336 and is encoded as a P frame in step 338. In a particular embodiment of the present invention, when the comparison in step 330 determines that differences between encoding the frame as an I or P frame are below a certain threshold, the frame is encoded as an I frame. Once encoded as an I frame or P

frame, the frame is inserted in the frame sequence at the appropriate position.

Which sets of frames undergo the process of determining whether to encode the frame as an I or P frame may be varied and may depend on the I, P, B, super-B, etc. frame pattern to be employed. Accordingly, the number of frames positioned between the frames that are processed may also be varied.

In one variation, every frame in a frame sequence may be considered to be a potential I or P frame and processed. Alternatively, only selected frames may be processed. For example, referring back to Figure 2B, only frames 4, 7, 10, 13, and 16 are processed in that particular embodiment.

As will be described herein, different numbers of B frames and super-B frames may be employed. Accordingly, the number of frames positioned between the frames that are processed to determine whether to encode them as I or P frames will depend on the types of frames to be employed (e.g., whether B, super-B are employed) as well as the I, P, B, super-B, etc. frame pattern to be employed. Optionally, B and even super-B frames may be processed to evaluate whether to encode those frames as I or P frames.

In one particular embodiment, all potential I/P frames in the sequence are processed unless an I frame has not been encoded for a predetermined number of frames. This is desirable when multiple channels of video are being transmitted. Since decoding of the next channel cannot occur until an I frame is encountered, it may be desirable to limit the time between I frames, for example within a half second.

According to an aspect of the invention, encoding of an I frame (for example at step 304 of the method of Figure 3A or at step 334 of the method of Figure 3B) may employ a whole frame encoder such as the MPEG-4 Textual Encoder or an encoder substantially similar to JPEG 2000. JPEG 2000 is a wavelet-based encoder. Employing a JPEG 2000 encoder to encode frames in a frame sequence according to this

aspect of the invention may provide a better compression and higher quality for the compressed image than other methods, including, for example, the discrete cosine transform which is conventionally used in MPEG-4.

i. Rate Control in Encoding Decisions

It is desirable that the quality of different frame types that are processed and evaluated be comparable when determining whether to encode a frame as an I or P frame so that relative coding efficiency can be fairly compared.

As described above, encoding decisions may be performed based on one or more cost functions that assess the efficiency of the encoding process according to various frame types. Such cost functions were previously discussed in connection with frame encoding format selection and the embodiments of Figures 3A and 3B. In addition to encoding efficiency, however, it is also desirable to consider the fidelity and accuracy of the encoding process. This is because a higher degree of compression of a particular frame may generally be achieved at the expense of the accuracy of the encoding process. Lossy image compression, for example, may generally achieve a higher compression rate by eliminating parts of the original data, and therefore distorting the original image. Consequently, in the embodiments of Figures 3A and 3B, it is desirable to establish a quality baseline for the different frames under consideration to provide a common framework for evaluating the efficiency of the encoding process.

An embodiment of the invention provides a method for rate control based on a normalization of the quantization step used to encode various frame types. Quantization is a stage performed during the process of encoding image frames compressed according to aspects of the present invention. For example, a rate distortion optimizer system 800 illustrated in Figure 8A and further described below, which may be employed to encode various frames considered during scene change

selection, comprises a quantizer 808. The quantizer 808 acts as an encoding stage for frames processed by the rate distortion optimizer system 800 and helps encode these frames. Quantization employs a certain step size to encode certain pixel coefficients determined during frequency transformation. As the step size of the quantization process increases, the compression factor of the corresponding frame may increase as well, but the fidelity of the encoded image with respect to the original image may decrease.

According to a particular embodiment of the invention, the quantization steps used to encode an I frame and a P frame during frame encoding decisions are normalized based on the following formula:

$$Q_I = \frac{Q_P}{K},$$

where Q_I is the quantization step used to encode the I frame, Q_P is the quantization step used to encode the P frame, and K is a constant. In a particular implementation, K is 120.

Normalization of the quantitization step may be employed in connection with scene change detection described in regard to Figure 3C to provide a baseline quality measure for the frames processed. Step 342 of the method illustrated in Figure 3C may be performed by the embodiment of Figure 3A and/or by the embodiment of Figure 3B. Normalization of the quantitization step may be implemented at steps 304 and 308 of the method illustrated in Figure 3A to normalize the quality of the encoded I and P frames whose sizes are compared at step 312. Analogously, the present method for rate control may be performed as part of steps 324 and 326 of the method illustrated in Figure 3B to normalize the quality of the encoded I and P frames whose cost functions are compared at step 330.

C. Scene Change Detection Based On Frame Type Encoding

A scene change occurs when, in the course of a video sequence, the global characteristics of the video image change. For example, when a change of scenery occurs suddenly in a video sequence such that within the span of at least two video frames the image changes from depicting an indoor view to depicting an outdoor image, a scene change may occur.

The video frames following a scene change tend to be uncorrelated with frames preceding the scene change. Consequently, encoding a P frame following the scene change by referring to an I frame or P frame preceding the scene change is inefficient and may result in errors. Further, since in such a situation encoding of subsequent P frames and B frames may rely on a potentially inefficiently and incorrectly encoded P frame, these subsequent P frames and B frames may also be inefficiently- or incorrectly-encoded. It is thus desirable to detect scene changes in order to facilitate efficient coding.

A further application of the decision logic for determining whether to encode a particular frame as an I frame or a P frame is for detecting a scene change in a video sequence. Unlike existing techniques that require complex data processing or provide relatively-inaccurate results, the decision logic of whether to encode a particular frame as an I frame or a P frame based on coding efficiency, optionally with a cost function, can be efficiently used to make scene change determinations.

Figure 3C shows an embodiment of a method for scene change detection based on a decision to encode a frame as an I frame or a P frame. According to the embodiment, a scene change is detected when a corresponding frame is encoded as an I frame. In the method of Figure 3C, a frame to be encoded is selected from the video sequence in step 340. In one embodiment of the present invention, the selection of a frame in step 340 is substantially similar to the selection of a frame

in step 322 of Figure 3B or in step 302 of Figure 3A. The frame selected in step 340 may be a partially or fully encoded MPEG-2 frame.

In step 342, a decision is made regarding the format in which the frame selected in step 340 is to be encoded. In one embodiment, the decision to encode a frame as an I frame or a P frame in step 342 is made as described in conjunction with the embodiments of Figures 3A or 3B. If the decision in step 342 is to encode the frame as an I frame, detection of a scene change is declared in step 344. If the decision in step 342 is to encode the frame as a P frame, the embodiment of Figure 3C declares in step 346 that no scene change has been detected. Otherwise, if the decision in step 342 is not to encode the frame as either an I or a P frame, the embodiment of Figure 3C declares in step 348 that no scene change has been detected.

An advantage of basing scene change detection on a decision whether to encode a frame as an I frame or a P frame as disclosed herein is that scene change detection logic may be simplified. For example, if the method of Figure 3A is employed to select an encoding format for frames in a frame sequence, scene detection is performed with little overhead beyond the data processing required for regular frame encoding. More specifically, since the embodiment of Figure 3A not only selects an appropriate format to encode the selected frame, but also encodes the frame in that format, identification of a scene change in step 344 may rely implicitly upon the result of the process of Figure 3A. Since the embodiment of Figure 3A both decides whether a particular frame should be encoded as an I frame or a P frame and also encodes the frame in the appropriate format, identification of a scene change according to an aspect of the present invention does not require additional specific complex data processing. It is noted that the process of Figure 3B as well as other processes for determining whether to encode a frame as an I frame or a P frame may also be used for determining scene changes.

Figure 3D illustrates a series of frame sequences where scene changes may be detected according to the embodiment of Figure 3C. Sequence 350 only includes an I frame in frame 1 and thus does not evidence a scene change during the sequence of frames shown. Frames 4 and 7 are illustrated as possible I or P frames. Upon processing in accordance with aspects of the present invention, a scene change may be detected in sequence 350 between frames 3 and 4, and frame 4 is consequently re-encoded as an I frame, as illustrated in sequence 352. Upon further processing, if no scene change is detected between frames 6 and 7 in the sequence 352, frame 7 may then be encoded as a P frame, thereby producing sequence 354.

D. Super-B Frames

In one embodiment of the present invention, a new type of frame is utilized, denoted herein as a Super Bi-directionally predicted frame ("SB" frame). Unlike B frames which may be encoded in reference to I and/or P frames, SB frames may be encoded in reference to I, P and/or B frames. The use of SB frames allows for a higher level of compression than using I, P and B frames alone. As a result, additional frames may be used to encode video.

An SB frame may reference preceded or subsequent I, P and/or B reference frames in the frame sequence. For example, a particular SB frame may be encoded in reference to a preceding I frame and a subsequent I frame; a preceding I frame and a subsequent P frame; a preceding I frame and a subsequent B frame; a preceding P frame and a subsequent I frame; a preceding P frame and a subsequent P frame; a preceding P frame and a subsequent B frame; a preceding B frame and a subsequent I frame; a preceding B frame and a subsequent P frame; or a preceding B frame and a subsequent B frame.

Figure 3E illustrates a sequence of frames compressed according to the present invention where SB frames are employed. The frame sequence of Figure 3E comprises I, P, B and SB frames. Frames 2, 4,

6, 8, 10, 12, 14 and 16 are illustrated as being SB frames. Frames 7 and 13 may be either I frames or P frames, depending on various decisions made during the encoding process as previously described in connection with Figures 2B and 3A-C.

SB frames may be encoded in reference to frames at various positions within the frame sequence. In one embodiment of the present invention, an SB frame is encoded in reference to only the nearest preceding I, P or B frame and the nearest subsequent I, P or B frame. For example, in Figure 3E, SB frame 4 may be encoded with respect to B frame 3 and B frame 5.

In another embodiment, an SB frame may be encoded in reference to I, P and/or B frames that are not the nearest preceding or following I, P and/or B frames. For example, SB frames may be encoded with respect to frames that are separated by one, two, three or more other frames. In this case, for example, SB frame 4 may be encoded with respect to I frame 1 and/or I or P frame 7.

Coding decision logic for encoding SB frames is used to control how many preceding and subsequent frames are employed to evaluate how to encode a given SB frame. All or only some of the frames evaluated may actually be used to encode the SB frame.

The coding decision logic may evaluate 0, 1, 2, 3, 4 or more preceding frames and 0 or 1 or more subsequent frames. Because SB frames are being encoded, the preceding and subsequent frames may be I, P or B frames. For example, in Figure 3E, coding decision logic may be designed to evaluate encoding SB frame 12 with respect to B frame 11 only; I or P frame 13 only; B frame 11 and/or I or P frame 13; B frame 11 and I or P frame 13; frames 9 and/or 11 and B frame 9 and I or P frame 13; etc.

Preferably, the coding decision logic evaluates at least two frames in order to encode a SB frame. The at least two frames preferably include at least one preceding frame and at least one subsequent frame.

How a given SB frame is actually encoded will depend on how the coding decision logic is programmed and the relative coding efficiencies in view of the coding decision logic. In one variation, the number preceding and subsequent frames the coding decision logic uses to encode a given SB frame is static. In another variation, the number preceding and subsequent frames the coding decision logic uses to encode a given SB frame varies as a function of coding efficiencies. According to this variation, if the underlying video sequence is substantially static such that no significant changes are captured between frames 12 and 13, the coding decision logic may encode SB frame 12 exclusively with respect to P frame 13. If the underlying video sequence is substantially static such that no significant changes are captured between frames 11 and 12, the coding logic encode SB frame 12 exclusively with respect to B frame 11. Alternatively, changes in frames 11, 12, and 13 may make it desirable to encode SB frame 12 with respect to B frame 11 and P frame 13.

It is recognized that the complexity and data overhead of the compression process increases as the number of preceding and subsequent frames used to make coding decisions for SB frames increases (e.g., with respect to 3, 4, 5, or more frames in a frame sequence). However, the use of a larger number of frames may nevertheless result in improved compression depending on the characteristics of the underlying video sequence. In Figure 4A, for example, SB frame 10 may be encoded in reference to I frame 7, B frame 9, B frame 11 and P frame 13.

Use of SB frames may help achieve a higher degree of compression than possible with I, P and B frames. One reason for this enhanced compression performance is that SB frames are encoded with respect to more than one frame, which permits optimization of encoding by decreasing or eliminating data redundancy. The higher efficiency of SB frames is also attributed to the fact that unlike B frames, SB frames can reference B frames in addition to I and P frames.

Since SB frames may reference subsequent I, P or B frames in the frame sequence, decoding of SB frames may require buffering of reference frames at the decoder to reconstruct the original frame structure.

It is recognized that SB frames, because they can rely on B frames, may include a higher degree of distortion than encoding those frames with reference to only I or P frames due to the multiple reference stages involved (e.g., an SB frame relying on a B frame which is encoded with respect to a P frame is the result of at least three predictive stages, and each of these stages may introduce a certain degree of distortion). Nevertheless, the cost associated with this potentially higher degree of distortion can be counterbalanced by the higher level of coding efficiencies that can be achieved. For example, distortion arising from the use of SB frames is less of an issue when encoding relatively static sequences. By using SB frames to encode static portions of a video sequence, a high level of coding efficiency can be achieved.

The higher level of coding efficiency achievable using SB frames for static frame sequences is illustrated, by way of example, with reference to Figure 3E. If a video scene corresponding to frames 7-12 is static, frames 10, and 12 can be efficiently encoded as SB frames (e.g., encoding them as "same as B frame 9") without loss of fidelity relative to having to reference them relative to I/P frames 7 or 13.

E. Enhanced Frame Layers for Improved Compression

To provide increased flexibility and scalability in the encoding and transmission of video data, an aspect of the present invention provides one or more enhanced compression frame layers that may be multiplexed in the global data stream. The enhanced layer is encoded in reference to a base layer and comprises frames (e.g., SB, SSB frames)

that are compressed to a higher degree than the frames in the base layer (e.g. I, P or B frames). The enhanced layers are encoded in a modular manner such that the base layer may be transmitted and decoded independently of, and without any information from the enhanced layers.

Figure 4A illustrates a layer structure comprising an enhanced frame layer for improved data compression according to an embodiment of the present invention. Figure 4A illustrates a base layer 402 and an enhanced layer 404. The base layer 402 comprises three types of frames: I frames, P frames and B frames.

As illustrated in Figure 4A, the base layer 402 and the SB frames in the enhanced layer 404 are encoded in reference to frames in the base layer 402. For example, frame SB_1 from the enhanced layer 404 is encoded in reference to frames I_1/P_1 and B_1 from the base layer 402

As illustrated in Figure 4A, the frames in the enhanced layer 404 are encoded based on frames in the base layer 402. As a result, decoding of the frames in the enhanced layer 404 may require prior decoding and buffering at the decoder of corresponding frames in the base layer 402. For example, the frame SB_2 from the enhanced layer 404 is encoded in reference to frames I_1/P_1 and B_2 from the base layer 402. Decoding of the frame SB_2 in this case would require prior decoding and buffering of the frames I_1/P_1 and B_2, at least to the extent necessary to extract the corresponding reference information.

As previously discussed in connection with the embodiment of Figure 4A, SB frames may be encoded in reference to an arbitrary number of frames. Consequently, although each of the frames in the enhanced layer 404 are illustrated as being encoded with respect to three frames in the base layer 402, the frames in the enhanced layer 404 may be encoded in reference to one or more base frames. For example, frame SB_1 may be encoded solely based on frame I_1/P_1, while frame SB_3 may be encoded in reference to frames I_1/P_1, B_1, B_2 and I_3/P_3 from the base layer 402.

Introduction of SB frames may lead to increased distortion in the encoded signal. However, since frames in the base layer do not reference frames in the enhanced layer, and since frames in the enhanced layer are not encoded with respect to other frames in the enhanced layer, distortion that may exist in the enhanced layer is not propagated through the frame sequence. Generally, the increased compression achieved using an enhanced layer comprising SB frames can be made to outweigh any disadvantage caused by introduction of a corresponding amount of distortion.

Coding decision logic may be employed according to aspects of the present invention to evaluate the balance between I, P, B and SB frames in the frame sequence. This balance is based on the bandwidth available as well as the amount of distortion the use of an enhanced layer introduces.

The ability for the base layer to be decoded independently of the enhanced layer provides increased flexibility and robustness in the encoding and transmission of video data.

A sequence having I, P and B frames in a base layer can be decoded with or without SB frames in the enhanced layer. As illustrated in Figure 4A, SB may frames exist in an enhanced layer which depend on frames in the base layer to be decoded. However, the frames in the base layer are not dependent on the frames in the enhanced layer (e.g., a SB frame) to be decoded. This allows the coded sequence to be temporally scalable. More specifically, the encoder may elect not to transmit portions of the enhanced layer for bandwidth considerations without disrupting transmission or decoding of the base layer. Accordingly, SB frames in an enhanced layer may be transmitted intermittently where possible to augment the video sequence being transmitted, without disrupting the video sequence when it is not feasible to transmit the enhanced layer.

It should be noted that the decoder may optionally elect to ignore SB frames in the enhanced layer, even if they are transmitted. In such

instances, the decoder would utilize I,P or B frames in the base layer to construct the missing frames. Hence, the independent decodability of I, P, and B frames relative to SB frames means that that transmission of SB frames does not make it necessary for a decoder to be able to use the SB frames.

In some embodiments, an encoder capable of producing a variable rate bitstream may use information regarding the bandwidth of the transmission channel to adjust its data transmission rate dynamically. If there is insufficient bandwidth, the encoder may use only the base layer to transmit the video sequence. An increase in the available bandwidth may result in the introduction of the enhanced layer to increase the quality of the signal. Introduction of the enhanced layer results in an increase in the frame rate.

In one embodiment, the transmission of enhanced layer(s) instead of the base layers in order to save on bandwidth demands is controlled by determining an amount of bandwidth available; determining an amount of bandwidth required to transmit the video data with or without the enhanced layer(s); and transmitting the video data with or without the enhanced layers based on a relationship between the determined amount of bandwidth available and the amount of bandwidth required to transmit the video data with or without the enhanced layer(s).

The amount of bandwidth available can be determined with different frequencies, depending on the application. For example, the available bandwidth may be monitored continuously, periodically at a fixed interval (which may be altered), or in response to some other signal or event.

The impact that using one or more enhanced layers have on the resulting quality of the transmitted images can vary depending on the content of the images and the level of enhanced layer used. Accordingly, it may be desirable to monitor the relative quality cost of using different enhanced layers. For example, when multiple channels are being broadcast simultaneously, the quality impact for each channel

of utilizing enhanced layers can be evaluated so that enhanced layers are employed for the channels where quality is less impacted by using an enhanced layer. As a result, a desirable balance between bandwidth requirements and image quality can be dynamically determined for the transmission of a plurality of channels.

The discussion regarding Figure 4A refers to the use of a single enhanced layer to improve the level of compression of video or multimedia data. According to an aspect of the invention, one or more additional enhanced layers may be employed to further improve the compression. Such additional enhanced layers are substantially similar to the enhanced layer described in connection with Figure 4A, but are encoded in a hierarchical manner such that each particular enhanced layer relies on one or more reference layers.

Figure 4B illustrates a layer structure comprising two enhanced frame layers for improved data compression according to an embodiment of the present invention: an enhanced layer 1 (416) comprising SB frames and an enhanced layer 2 (376) comprising SSB frames. SSB (Super SB) frames have been previously described in connection with the embodiment of Figure 3E. Figure 4B also shows a base layer 412 and a B frame layer 414 analogous to the base layer 402 and, respectively, the B frame layer 404 from Figure 4A. The enhanced layer 1 (416) comprises SB frames encoded based on frames in the base layer 412 and is substantially identical with the enhanced layer from Figure 4A.

Encoding of additional enhanced layers like the enhanced layer 2 (376) employs substantially the same principles as previously described in connection with the encoding of a single enhanced layer and Figure 4A. More specifically, each additional enhanced layer is encoded in a modular manner based on one or more reference layers. Decoding of the reference layers may be performed without any information from the dependent enhanced layer. Encoding of any additional enhanced layer

may introduce additional computational complexity but may provide improved data compression.

2. Video Compression System

The compression and encoding of a video sequence in accordance with an aspect of the present invention comprises a series of steps whereby video or multimedia data is compressed and encoded into various types of frames. Figure 5 illustrates a schematic diagram of a system for compressing video that comprises various aspects of the present invention. According to Figure 5, an input signal is processed by a number of successive subsystems to produce an encoded and compressed variable rate bitstream.

It is noted that the various inventions described herein relate to novel aspects that may be each independently incorporated into the different subsystems. These different inventions may be used in video compression systems that depart from the overall system shown in Figure 4 and still remain within the scope of the present invention.

The input signal may be any type of data stream comprising a sequence of images, including video or film, a multimedia data stream, or a television signal. In one case, the input signal is a television signal formatted in accordance with the National Television Standard Committee (NTSC) standard. Alternatively, the input signal may be a television signal formatted according to the Phase Alternating Line (PAL) or the Sequential Couleur Avec Memoire (SECAM) standards.

In addition to analog, modulated signals, the input signal may also be a digitally-encoded data stream. For example, the input signal may be a digital television signal encoded in accordance to a standard promoted by the Advanced Television Systems Committee. In a particular case, the input signal may be a High Definition Television signal. Alternatively, the input signal may be an MPEG data stream. In one embodiment of the present invention, the preprocessing system 502 is designed to operate upon analog signals. In that case, if the input

signal is a digitally-encoded data stream, a conversion stage is introduced prior to the preprocessing system 502 to convert the digital data stream to an analog signal. Alternatively, the digitally-encoded data stream may be processed by a decoder to produce a digital video signal compatible with the motion estimation system 504 and may bypass the preprocessing system 502 to enter directly into the motion estimation system 504. In a particular case, an MPEG-2 data stream may be decoded and may be fed directly into the motion estimation system 504.

According to Figure 5, the input signal is initially digitized and filtered by the preprocessing system 502 to produce a sequence of digital frames. The preprocessed signal is then fed into the motion estimation system 504, which employs various coding models to produce a set of motion vectors estimating relative changes of objects within the frames. The signal is then processed by a coding decision system 506 that relies on the information derived by the motion estimation system 504 to compress and encode each individual frame at a macroblock level. The coding decision system 506 comprises a mode processing subsystem 508, a rate distortion optimization subsystem 510 and an encoding selection subsystem 512. The mode processing subsystem 508 applies various encoding modes to each macroblock of each frame being encoded to produce a set of candidate encoding formats. The rate distortion optimization subsystem 510 processes each of the candidate encoding formats for each macroblock to determine a set of corresponding cost functions. The encoding selection subsystem 512 compares the cost functions to identify a preferred encoding format for each macroblock. Data compressed in accordance to the preferred encoding format is then transmitted by the coding decision system 506 as an output encoded bistream.

The systems and subsystems illustrated in the embodiment of Figure 5 are described in further detail below.

A. Motion Estimation

Motion estimation may be employed to determine the motion of image objects as captured between different frames. The positions of various objects within an image generally change in a progressive manner from frame to frame. Local motion estimation and global motion compensation represent two approaches to motion estimation that may be employed to detect changes in the position of various objects between frames and to efficiently encode the frames.

Local motion estimation may be employed to estimate the direction and magnitude of motion of a particular image object between an encoded reference frame and a dependent frame. This data may then be employed to efficiently encode the dependent frame by referencing the encoded representation of the object in the reference frame. Specifically, once a particular object is encoded in a reference frame, the representation of that object in a dependent frame may be encoded employing motion estimation to identify the prior position of the object within the reference frame and referencing the encoded representation of the object. If the representation of the object in the dependent frame is different from the representation of the object in the reference frame even after the motion of the object is taken into account, the differences may be also encoded as part of a prediction error signal.

Global motion compensation ("GMC") represents a second type of motion estimation that may efficiently encode image frames by compensating for global, frame-level image movement induced by effects such as translation, zoom and angular rotation. GMC frame encoding is normally performed in conjunction with local motion estimation as it usually relies on information derived during local motion estimation.

The relative inter-frame motion of image objects may be described in a two-dimensional reference frame using motion vectors. A motion vector is a two-dimensional vector that records the spatial offset between the coordinate position of an object in a dependent frame and

the coordinate position of the object in the reference frame. A motion vector is normally represented by two components, {x, y}, which indicate spatial offsets along the x and y axis, respectively. The components of a motion vector may be integer numbers when displacement is measured at pixel level, or may be non-integer numbers when motion is recorded at a higher resolution that relies on sub-pixel coordinates. Motion vectors generally provide a framework for establishing reference relationships between spatial image areas of different frames.

Figure 6A illustrates a flow diagram describing an embodiment of motion estimation logic that may be used in conjunction with an aspect of the present invention. In one implementation, the motion estimation system 600 from Figure 6A is the motion estimation system 504 illustrated in Figure 5. The motion estimation module 6A may receive a preprocessed signal from the preprocessing system 502 from Figure 5 and may analyze this signal to derive information that may be used subsequently to select an efficient encoding method.

i. Macroblocks

According to an embodiment of the invention, each frame being encoded is partitioned into macroblocks prior to being processed by the motion estimation system 600. This partitioning process may be accomplished by the preprocessing system 502 of Figure 5. Each macroblock comprises a rectangular two-dimensional array of image pixels. Partitioning an image in pixel macroblocks facilitates image analysis and manipulation during the encoding process by, among others, increasing the likelihood of spatial correlation among samples in the image block.

The optimal size of macroblocks depends on balancing a number of factors. Decreasing the size of the macroblocks may increase the computational complexity by increasing the number of blocks that have to be processed. However, a smaller size for the macroblocks also provides advantages, including reducing the amount of data that is

processed for each block and increasing the accuracy with which individual areas of the picture may be identified. In existing MPEG implementations, macroblocks have size of 16x16 image pixels. Matrices of pixels with dimensions of less than 16x16 are generally denoted as blocks. In one embodiment of the invention, blocks have size of 8x8 image pixels.

In one embodiment of the invention, the image is divided into blocks of 4x4 image pixels. Advantages of reducing the size of blocks to 4x4 pixels include increased flexibility in manipulation of data at macroblock and pixel levels, with a corresponding improvement in the accuracy and degree of compression of the underlying video stream.

Each macroblock pixel has a number of attributes that may serve as a basis for encoding the macroblock, including spatial attributes and graphical attributes. The spatial attributes of a pixel include the location of the pixel within the respective macroblock or frame. This spatial location may be measured either in absolute or relative coordinates. In absolute coordinates, the location of a pixel may be recorded in a two dimensional Cartesian coordinate system, where the first pixel in a particular macroblock has coordinates {0, 0}. Alternatively, the location of a pixel may be recorded in relative coordinates, with respect to a reference point within the respective frame or macroblock. Identification of a pixel in relative coordinates is typically performed using a motion vector, which records {x, y} offset values with respect to a particular point in the respective frame or macroblock. Motion vectors and relative coordinates provide a convenient method for encoding macroblocks in reference to other macroblocks.

The graphical attributes of pixels typically include luminance and chrominance information. Luminance and chrominance may be represented in various formats. An embodiment of the invention preferably operates on images represented in the YUV color space (Y Cr Cb). For this embodiment, images encoded in other formats, including the RGB format, are initially converted to the YUV format. If appropriate,

this conversion may be performed by the preprocessing system 502 in the embodiment of Figure 5.

In a 4:2:0 format, the graphical information associated with a particular macroblock is organized in six blocks: four luminance blocks and two chrominance blocks. The methods disclosed herein are not restricted to 4:2:0 formats, but may also be applied to other formats including, for example, 4:2:2 and 4:4:4 formats. In a 4:2:2 format, the graphical information associated with a particular macroblock is organized in eight blocks: four luminance blocks and four chrominance blocks. In a 4:4:4 format, the graphical information associated with a particular macroblock is organized in twelve blocks: four luminance blocks for Y channel and four chrominance blocks for U and V channel each. For a macroblock with pixel dimensions of 2Mx2M, each of the four luminance blocks and each of the two chrominance blocks in the 4:2:0 format has dimensions of MxM pixels. For typical MPEG applications, M is 8. In a particular implementation of the invention, M is 8, such that macroblocks have dimensions of 16x16 pixels, and the luminance and chrominance blocks are each 8x8 pixels.

The chrominance and luminance data associated with a particular macroblock is typically encoded in 24 bits per pixel: 8 bits for the Y luminance information, 8 bits for the U chrominance information, and 8 bits for the V chrominance information. The YUV format is then subsampled to reduce the number of bits per pixel while limiting the impact on the qualitative aspects of the image. Although all the luminance information is retained, chrominance information is subsampled 2:1 in both the horizontal and vertical directions.

This subsampling process has a limited impact upon the quality of the image because human vision is more sensitive to luminance than to chrominance information. Subsampling is a lossy step because information regarding chrominance is selectively eliminated. In the case of 24 bit-per-pixel RGB images, for example, subsampling reduces the information necessary to represent a macroblock to 12 bits per pixel,

which provides a 2:1 compression factor. Encoding of a macroblock in reference to a different macroblock typically requires encoding of luminance and chrominance information for each pixel in the macroblock. Consequently, decreasing the number of bits used to record luminance or chrominance improves the efficiency of the encoding process.

The motion estimation 600 illustrated in Figure 6A comprises a local motion estimation module 602 and a global motion estimation module 604. As illustrated in Figure 6A, the motion estimation system 600 processes a current frame together with a number of reference frames to generate a set of data that may be used to encode the current frame. The current frame is partitioned into N macroblocks, where the number N depends on the total number of pixels in the image frame and the size of the macroblocks. The N macroblocks are denoted as a series $\{MB_1, MB_2, \ldots, MB_N\}$.

Partitioning a frame that is being encoded into macroblocks may be achieved by mathematically dividing the frame into contiguous macroblocks of identical size. In this partitioning scheme, boundaries of macroblocks may be determined unequivocally based on a relative position within the frame. In contrast, identification of macroblocks in reference frames corresponding to macroblocks in the frame being encoded does not normally result in partitioning of the reference frames into contiguous macroblocks. Instead, it often happens that a macroblock in a reference frame selected to act as a predictor for a particular macroblock being encoded is located at an arbitrary position within the reference frame, possibly overlapping with one or more other macroblocks within that reference frame. In other cases, when motion vectors are defined with a higher resolution, it is possible that macroblocks in the reference frames are defined in non-integer pixel coordinates, such that the boundaries of particular reference macroblocks fall between the physical pixels of the image pixel matrix.

The local motion estimation module 602 processes the N macroblocks to derive a set of R motion vectors, denoted as $\{MV_1, MV_2, \ldots, MV_R\}$. The local motion estimation module 602 produces at least one motion vector for each macroblock. Consequently, the number R of motion vectors will be at least as large as the number N of macroblocks. The motion vectors determined by the local motion estimation module 602 are subsequently used by the coding decision system 608 to select a preferred method for encoding the corresponding frame.

The motion vectors are also made available to the global motion estimation module 604, which may use them to determine the nature and magnitude of image object transformations that occur within the current frame with respect to other frames. Both the global motion estimation module 604 and the local motion estimation module 602 evaluate various methods for encoding the respective macroblock and provide relevant encoding data to the coding decision system 608 to select a preferred method for encoding the macroblock in the current frame.

(ii). Local Motion Estimation

The local motion estimation module 602 processes a current frame together with a number of reference frames to produce a set of motion vectors and other data that may be employed to encode the current frame. One goal of the local motion estimation module 602 is to identify one or more reference macroblocks that exhibit a relatively-high degree of similarity with the current macroblock being encoded. This similarity may be determined on a pixel-by-pixel basis by comparing various pixel attributes of the current and reference macroblocks, including pixel luminance.

Figure 6B illustrates a flow chart of a local motion estimation process that may be used in conjunction with motion estimation according to an aspect of the present invention. In one implementation, the local motion estimation process 630 from Figure 6B is performed by

the local motion estimation module 602 illustrated in Figure 6A. In the embodiment of Figure 6B, the local motion estimation process 630 processes a current image frame and a set of reference frames to estimate the direction and magnitude of motion of various macroblocks/blocks within the current frame.

At step 610, a particular frame to be encoded is selected from the frame sequence. This frame will ultimately be encoded as an I, P, B, SB, or SSB frame, depending on the decision made by the coding decision system 634. In addition to the frame being encoded, a set of reference frames is also selected at step 610. One or more of these reference frames will serve as a basis for encoding the current frame.

(a) Selection of Reference Frames

The selection of reference frames at step 610 seeks to identify one or more reference frames that may be advantageously used as a basis for encoding a current frame. Selection of reference frames may be based on various criteria, including, for example, processing complexity, signal fidelity or compression performance.

In one case, selection of reference frames at step 610 seeks to identify reference frames that exhibit a high degree of similarity with respect to the current frame. This could result in a higher degree of compression for the current frame and may ensure a more accurate representation of the original image.

Selection of relatively similar image frames may be accomplished based on various criteria. For example, reference frame selection logic employed at step 610 may decide that a particular frame is likely to be transmitted again at a subsequent time and may therefore select that frame for encoding and transmission to the decoder. In one implementation, the reference picture selection logic may scan subsequent frames of the video sequence, beyond a particular reference frame being processed, to identify a substantially-similar frame. If such

a frame is identified, the reference picture selection logic may select it as a reference frame.

In one case, digital signal processing techniques may be employed to determine the degree of correlation between the current frame and candidate reference frames. In another case, a number of contiguous preceding and/or subsequent frames from the frame sequence may be buffered and utilized as candidate reference frames for the current frame without any further selection. Generally, as the number of candidate reference frames increases, the reference frame buffers, the computational complexity and the memory necessary to process these reference frames also increases. Depending on the properties of the underlying video sequence, however, increasing the number of candidate reference frames may help increase the compression and the quality of the encoding process to counterbalance any corresponding disadvantages.

Selection of reference frames is further described below in connection with the multiple hypothesis macroblock encoding mode and the embodiment of Figure 7C.

(b) **<u>Hierarchical Local Motion Estimation</u>**

Upon selecting a frame to be encoded and a set of candidate reference frames at step 610, the local motion estimation process progresses to step 611 where both the current frame and the candidate reference frames are downsampled. The downsampling process reduces the resolution of the frames and is accomplished using an averaging filtering process.

In one embodiment of the present invention, the downsampling process at step 611 is performed recurrently three times to obtain a set of four layers with a progressively lower resolution. In a first downsampling stage, rectangular macroblocks with an original size of 16x16 pixels are reduced to sizes of 8x8 pixels. In successive stages, the macroblocks are further donwsampled to sizes of 4x4 and 2x2

pixels. Each successive downsampling stage provides a corresponding layer: 16x16 macroblocks are part of a layer 1, 8x8 macroblocks are part of a layer 2, 4x4 macroblocks are part of a layer 3, and 2x2 macroblocks are part of a layer 4. Each of these layers will provide a basis for progressive refinement of the local motion estimation process in subsequent steps.

Each downsampling stage may be accomplished in two main phases: an antialiasing low pass filtering phase and a subsampling phase. In a first antialiasing low pass filtering phase, the luminance attribute of each set of two contiguous pixels is averaged to produce an average luminance value. In the second subsampling phase, the two contiguous pixels are replaced by a single pixel with a luminance equal to the average luminance value determined in phase 1. As a result, macroblocks in inferior layers are replaced by smaller, filtered macroblocks in superior layers.

In an alternative embodiment, more than three subsampling stages may be performed. This may be the case, for example, if the size of the macroblocks in layer 1 is larger than 16x16. In yet another embodiment, instead of applying the antialiasing low pass filtering process to luminance of pixels, the filtering process may be based on chrominance or on other pixel attributes.

Once the frames are downsampled in step 611, a first 2x2 macroblock is selected in the current downsampled frame at step 612. Steps 613 and 614 will subsequently identify a set of 2x2 macroblocks in the candidate reference frames to serve as effective reference macroblocks.

At step 613, a first candidate reference frame is retrieved and a downsampled search range ("DSR") is defined within the candidate reference frame. The downsampled search range DSR identifies an area within which the local motion estimation process 614 will exhaustively search to locate a possible reference macroblock for the current macroblock. In one embodiment, the search range consists of a

rectangular pixel matrix with a side length of 2*DSRL, where DSRL is provided by the following expression:

$$DSRL = \frac{64\,d}{2^{L-1}}, \qquad (3)$$

where d represents the number of frames between the current frame and the respective reference frame in the frame sequence and L is the number of the current subsampling layer. For layer 4, L = 4.

At step 614, the system searches exhaustively the search range for one or more 2x2 macroblocks that exhibit a relatively-high degree of similarity with the current downsampled macroblock. The degree of similarity between the candidate reference macroblocks and the current macroblock may be assessed based on a cost function. In a particular embodiment of the invention, the cost function is a Sum of Absolute Differences ("SAD") function that determines a cumulative sum of absolute differences in luminance values between pixels in the current macroblock and corresponding pixels in the candidate reference macroblock. In one implementation, the SAD function is expressed as follows:

$$SAD = \sum_{i,j} a(i,j) \bullet \left| Diff(i,j) \right|, \qquad (4)$$

$$\text{where } a(i,j) = \mathrm{w(i)} \bullet \mathrm{w(j)}, \quad \text{and } \mathrm{w(k)} = \frac{1}{2}\left(1 + \cos\frac{\pi(2k - N + 1)}{N}\right), k = 0,1,2 \ldots N .$$

In equation (4), Diff(i,j) represents the difference between the luminance values of corresponding pixels in the current macroblock and the candidate reference macroblock. Further, N represents the size of the macroblock in pixels. In layer 4, N = 2.

The SAD function expressed in equation (4) provides a measure of the pixel-by-pixel differences between the current macroblock and a

potential reference macroblock. A lower value of the SAD indicates a higher degree of similarity between the two macroblocks. To identify a set of preferred reference macroblocks, the exhaustive search at step 614 constructs all possible macroblocks within the search range provided by equation (3) and determines SAD values for each of these macroblocks. The system then ranks the macroblocks based on their corresponding SAD values and selects a set of macroblocks with the lowest SAD values.

In one embodiment, the system selects the four macroblocks with the lowest SAD values and determines a motion vector for each of these macroblocks. The resulting four macroblocks identify the best reference macroblocks in the corresponding reference frame. In alternative embodiments, the system may select more or less than four candidate reference macroblocks for each macroblock being encoded. In one particular case, the system selects eight candidate reference frames. The search for best reference macroblocks performed at steps 613 and 614 is then repeated within each candidate reference frame. As a result, for each candidate reference frame, the system derives a set of four motion vectors identifying the best reference macroblocks within that reference frame. Each of these sets of four motion vectors will be further processed in subsequent steps to ultimately derive a single best reference macroblock across all candidate reference frames for the macroblock selected in step 612.

An aspect of the present invention provides a method of hierarchical motion estimation where multiple candidate motion vectors are passed from a lower downsampled layer to a higher layer. By transmitting multiple motion vectors to a higher level, this method may help identify better reference macroblocks for a particular macroblock being encoded. As a result, the motion estimation process may be performed with a higher degree of accuracy, which may result in enhanced compression and increased fidelity of the encoded signal. Each of the motion vectors transmitted from the lower downsampled

layer serves as a basis for a reference macroblock search, thereby expanding the effective search area and increasing the probability of identifying a better-match reference macroblock.

In the local motion estimation process 630, the system passes multiple motion vectors from a lower layer to a higher layer at steps 615, 619, 623 and 626, which are described below. Additional instances of transmittal of multiple motion vectors between successive layers exist, for example, within steps 628 (refinement of motion vectors using a multiple pass process) and 629 (refinement of motion vectors using fractional pixel refinement). Steps 628 and 629 will be further described in connection with Figures 6C and 6E.

At step 615, the four motion vectors identified in step 614 are scaled in preparation for further reference macroblock searching in layer 3. Since in the embodiment of Figure 6B the dimensions of macroblocks in layer 3 are double the dimensions of macroblocks in layer 4 (i.e., 4x4 compared to 2x2), the motion vectors determined in step 614 are scaled by a factor of 2. More specifically, the x and y components of each motion vector determined in step 614 are multiplied by 2. The resulting scaled motion vectors will serve as a starting point for a new search for reference macroblocks.

At step 616, the system identifies a 4x4 macroblock in the current frame corresponding to the downsampled 2x2 macroblock selected in step 612. At step 617, the system then determines a new downsampled search range. The downsampled search range for layer 3 may be determined based on equation 3 or may be arbitrarily defined. Generally, a larger search range may help locate a better reference macroblock, but possibly at a higher computational expense. In one embodiment, the downsampled search range in layer 3 is a 10x10 square pixel matrix centered around the coordinates of each motion vector scaled in step 615. Since a total of four motion vectors corresponding to four candidate reference macroblocks were transmitted from step 615, a total of four search ranges are determined in step 617

for each 4x4 macroblock being encoded. Determination of search ranges at step 617 is analogous to the selection of a search range at step 613.

At step 618, the system employs a process analogous to the process described in connection with step 614 to identify a set of four motion vectors corresponding to four minimal-SAD macroblocks within each candidate reference frame. These motion vectors are then scaled by a factor of 2 in preparation for further processing within layer 2, analogously to the scaling process previously performed at step 615.

Steps 620, 621, 622 and 623 identify a set of four best-match reference macroblocks for an 8x8 macroblock in layer 2 employing processes and methods analogous with the ones described in connection with steps 616, 617, 618 and 619 from layer 3. The search range utilized at step 621 is defined as a 14x14 pixel matrix. An analogous set of processes and methods are further employed in layer 1 at steps 624, 625 and 626 to identify a set of four 16x16 minimum-SAD reference macroblocks. The layer 1 search range is defined at step 625 as a 22x22 pixel matrix. At step 626, the system produces a set of four motion vectors corresponding to four 16x16 best-match reference macroblocks. It is noted that other search ranges and block sizes may also be used.

At step 631, the block-level motion estimation module receives a set of four motion vectors corresponding to the current macroblock determined from step 626. In one embodiment, the system determines one or more motion vectors for each block within the current macroblock. Step 631 will be further described in connection with Figure 6D.

A decision module 627 determines whether all macroblocks in the current frame being encoded have been processed using the hierarchical block search process described above. If all macroblocks within the current frame have been processed, the system progresses to step 628 where the motion vectors previously derived are further refined.

Otherwise, the system returns to step 612 in layer 4 to process the next 2x2 downsampled macroblock.

(c) Block-Level Local Motion Estimation

An aspect of the present invention provides a method for subdividing a macroblock into smaller blocks and estimating the motion of the smaller blocks hierarchically from the motion of larger blocks. The smaller blocks may be rectangular or square. The motion estimation of the smaller blocks may be performed based on the hierarchical motion estimation process previously described in connection with the local motion estimation process 630 from Figure 6B. This method applies blocks within macroblocks, including 8x8 blocks.

Figure 6D illustrates a flow diagram for block-level local motion estimation according to an aspect of the present invention.

In the block-level motion estimation process 653 from Figure 6D, a current macroblock being encoded and a set of four corresponding reference motion vectors are received at step 654. The reference motion vectors correspond to reference macroblocks that may have been identified during hierarchical local motion estimation process, as previously described in connection with the embodiment of Figure 6B. The current macroblock is partitioned into a number of rectangular or square blocks at step 654. In one embodiment, the current macroblock is partitioned into four square blocks of equal size. At step 655, the system assigns to each of the four blocks the set of four motion vectors corresponding to the current macroblock. Each of these motion vectors will serve as an initial basis for further motion estimation for each of the four blocks.

At step 656, the system then determines a search range within the reference frame for each block of the current macroblock. The search ranges are defined as generally described in connection with the embodiment of figure 6B. In one embodiment, the search ranges for the

four blocks consist of 14x14 pixel matrices centered around the corresponding motion vectors.

At step 657, the system performs exhaustive searches within each search area to identify a set of four best-match reference blocks for each of the four current blocks. The search process may rely on a SAD cost function and is analogous with the exhaustive search process described in connection with the embodiment of figure 6B.

The system then outputs a set of best-match motion vectors corresponding to each block comprised in the current macroblock. Each of these motion vectors identifies a corresponding reference block in a corresponding reference frame. Additionally, the system also outputs a set of best-match motion vectors corresponding to reference macroblocks for the current macroblock. In one case, the system outputs four reference motion vectors for each block comprised in the current macroblock and four reference motion vectors corresponding to the current macroblock. As illustrated in Figure 6B, these motion vectors may be transmitted to a multiple pass motion vector refinement module for further processing at step 628.

(d) Multiple Pass Motion Vector Refinement

An aspect of the present invention provides a method of multi-pass reference motion vector refinement where reference macroblocks/blocks corresponding to macroblocks/blocks located proximally to a current macroblock/block being encoded are compared to the current macroblock/block based on a cost function to determine a preferred reference macroblock/block. Multiple pass motion vector refinement performs at a macroblock level (e.g., for16x16 macroblocks) and at a sub-macroblock level (e.g., for 8x8 blocks). If evaluation of the cost function indicates that a particular reference macroblock/block corresponding to a macroblock/block located proximally to the current macroblock/block is a better match than a candidate reference

macroblock/block corresponding to the current macroblock/block, the motion vector associated with the particular reference macroblock/block is substituted for the motion vector of the candidate reference macroblock/block and is selected as a basis for further motion estimation processing. This process is repeated multiple times for all macroblocks/blocks in a reference frame until no more motion vector substitutions occur. In one case, the number of repetitions is limited to an arbitrary number. This process may be performed using 16x16 macroblocks, larger macroblocks, or smaller blocks, including 8x8 blocks.

If all the 16x16 macroblocks in the current frame have been processed in the first stage of the hierarchical motion estimation process of the embodiment of Figure 6B, the decision module 627 provides a set of four motion vectors for each 16x16 macroblock and a set of four motion vectors for each 8x8 blocks within a macroblock in the current frame to a multiple pass refining system at step 628. Each of the four motion vectors identifies a best-match reference macroblock for the corresponding 16x16 macroblock in the current frame. Each of the four motion vectors identifies a best-match reference block for the corresponding 8x8 block in the current frame. The multiple pass refining system at step 628 processes each macroblock or block in the current frame to further refine its corresponding motion vectors.

Figure 6C illustrates a flow diagram for multiple pass motion vector refinement according to an aspect of the present invention. In one embodiment, the multiple pass motion vector refinement of Figure 6C is performed as part of step 628 comprised in the local motion estimation process of Figure 6B.

As illustrated in Figure 6C, the multiple pass process 640 receives a set of reference motion vectors corresponding to macroblocks and blocks in a current frame being encoded, refines these reference motion vectors, and outputs the refined motion vectors for further processing.

A current macroblock/block to be processed is initially selected at step 641. During subsequent passes, a contiguous macroblock/block is sequentially selected at step 641. The macroblock/block selected at step 641 and a set of reference vectors corresponding to that macroblock/block are received at step 642. The set of reference vectors received at step 642 may comprise the four reference motion vectors identified in step 631 of the local motion estimation process 630 from Figure 6B.

At step 643, a set of neighboring macroblocks located proximally to the current macroblock and/or a set of neighboring blocks located proximally to the current block are identified. In one case, the neighboring macroblocks/blocks are the four macroblocks/blocks that share a side with the current macroblock/block. In another case, the set of neighboring macroblocks/blocks may also comprise the four macroblocks/blocks that share a corner with the current macroblock/block. The number of neighboring macroblocks/blocks may vary from eight (for a macroblock/block located in the center to the reference frame) to three (for a macroblock/block located in a corner of the reference frame). In one case, the set of neighboring macroblocks/blocks may include macroblocks/blocks that are separated from the current macroblock/block by one or more macroblocks/blocks.

At step 644, the system receives a set of reference motion vectors corresponding to the neighboring macroblocks or a set of reference motion vectors corresponding to the neighboring blocks identified at step 642. For each neighboring macroblock, the system identifies one or more corresponding reference macroblocks (denoted "neighboring reference macroblocks"), as indicated by the corresponding motion vectors. For each neighboring block, the system identifies one or more corresponding reference blocks (denoted "neighboring reference blocks"), as indicated by the corresponding motion vectors. These neighboring reference macroblocks/blocks are received at step 646,

together with the reference macroblocks/blocks corresponding to the current macroblock/block identified at step 642.

The system then determines a cost function for each of the neighboring reference macroblocks/blocks identified at step 644 with respect to the current macroblock/block being encoded. These cost functions are denoted "neighboring reference cost functions." The system also determines cost functions for each reference macroblock/block identified at step 642 with respect to the current macroblock/block being encoded. These cost functions are denoted "reference cost functions." In one embodiment, each reference cost function and each neighboring reference cost function is determined based on the SAD function expressed in equation (4) and previously discussed in connection with step 614 of the local motion estimation process 630 from figure 6B.

At step 647, the system compares all reference cost functions and neighboring reference cost functions to identify one or more best-match reference macroblocks/blocks. The system retrieves the motion vectors corresponding to the reference macroblocks/blocks with the best-cost functions. In one embodiment, the system selects the motion vectors corresponding to the four macroblocks/blocks with the lowest SAD values.

At step 648, the system decides whether all macroblocks/blocks in the current frame have been processed. If additional macroblocks/blocks remain unprocessed, the system returns to step 641 where the next macroblock/block is selected to be processed. Otherwise, the system progresses to step 649 where it decides whether the previous pass resulted in any reference macroblocks/blocks being replaced by their respective neighbors. Any such change may indicate that previous motion estimation processing may have inaccurately identified reference macroblocks/blocks for the respective macroblock/block and/or for other macroblocks/blocks within the frame being encoded. Substitutions of motion vectors corresponding to

neighboring macroblocks/blocks as better reference motion vectors may propagate through the reference frame. Subsequent passes through the current frame may identify better matches for macroblocks/blocks in the frame currently being encoded.

If no changes were detected during the prior pass, the system outputs the resulting sets of four motion vectors corresponding to each macroblock/block in the present frame. In one embodiment, the system ceases further processing within the multiple pass process 640 upon completion of a certain number of passes, even if changes within the reference frame were detected in the previous pass. This may be appropriate when subsequent passes and additional processing may result in insufficient improvements in compression or accuracy of encoding. In a particular implementation, the system exits the multiple pass process 640 after 15 passes.

(e) Fractional Pixel Motion Vector Refinement

Fractional pixel refinement is a method for refining motion vectors to increase prediction accuracy to sub-pixel levels. Fractional pixel refinement may be performed at a macroblock level (e.g., for 16x16 macroblocks) or at a sub-macroblock level (e.g., for 8x8 blocks). Fractional pixel refinement helps improve the accuracy of the motion vectors derived during local motion estimation for the macroblock. In one embodiment, the macroblock is divided into smaller blocks, and individual motion vectors are determined and refined to fractional pixel accuracy for each block to encode each block individually based on one or more corresponding reference blocks.

Figure 6E illustrates a flow diagram for fractional pixel motion vector refinement according to an aspect of the present invention. In one embodiment, the fractional pixel refinement process 652 from Figure 6E is performed within step 629 of the local motion estimation process 630 from Figure 6B.

An aspect of the present invention provides a method for performing motion estimation with non-integer pixel precision where sub-pixel motion is estimated hierarchically from motion vectors corresponding to lower-precision layers. In one embodiment, this is achieved by progressively increasing the resolution of the reference frame and applying the hierarchical motion estimation process previously discussed in connection with the embodiment of Figure 6B. This method for performing motion estimation with non-integer pixel precision is further described below.

At step 658, the system receives a set of reference motion vectors identifying best-match reference blocks for the four blocks comprised in the current macroblock. In one embodiment, these reference motion vectors were determined during block-level local motion estimation at step 631 of the local motion estimation process 630 from Figure 6B. These motion vectors may all point into the same reference frame. Upon receiving the reference motion vectors, the system upsamples the corresponding reference frame. In one embodiment, the reference frame is upsampled by a factor of two. In that case, reference blocks corresponding to the current blocks expand from a size of 8x8 to a size of 16x16 pixels. One way to upsample the reference frame is through a process substantially opposite to the averaging filter employed during downsampling of reference frames at step 611 of the local motion estimation process 630 from Figure 6B. More specifically, the luminance value of each pixel in the 8x8 block is assigned to a group of two contiguous pixels in the 16x16 upsampled block.

Upsampling of a reference frame provides a convenient coordinate system for identifying increased-precision, sub-pixel object motion. In a particular case where the upsampling factor is two, for example, block motion in the reference frame may be tracked with a precision of ½ a pixel.

At step 659, the system scales the motion vectors to maintain accurate pointing within the upsampled reference frame. In the case when the upsampling factor is two, the system also scales the motion vectors by a factor of two. At step 660, for each of the current blocks being encoded, the system identifies four upsampled reference blocks in the upsampled reference frame. Additionally, the system identifies four upsampled reference macroblocks corresponding to the current macroblock.

The system then progresses to step 661 where it determines a set of four search ranges for each of the blocks in the current macroblock and for the macroblock itself. Each of the four search ranges corresponding to a current block or macroblock is centered around the corresponding motion vector and exceeds the size of the corresponding reference block or macroblock by 3 pixels in all four directions of the upsampled reference frame. The system then performs a SAD-based exhaustive search within each search area at step 662 to identify a set of four best-match reference blocks for each of the current blocks and a set of four reference macroblocks for the current macroblock.

The system then proceeds to step 664 where it decides whether the motion estimation process has been performed with sufficient accuracy. In one embodiment of the invention, motion estimation is performed with a sub-pixel accuracy of 1/8. To achieve this accuracy, the system returns to step 658 from step 664 to repeat the upsampled motion estimation process previously described in connection with steps 658-662. In this iteration, the search range at step 661 is centered around the corresponding motion vector and exceeds the size of the corresponding reference block or macroblock by 3 pixels in all four directions of the upsampled reference frame.

In one embodiment, prior to exiting the local motion estimation process, the system performs a SAD-based final ranking of reference blocks and reference macroblocks to identify a single best-match

reference macroblock for each of the current macroblocks and a single best-match reference block for each of the four blocks in the current macroblock. This final ranking process may be performed only within particular reference frames, or may be generalized over all reference frames to identify absolute, multiple-reference-frame-level best matches.

(iii). Global Motion Estimation

Global motion compensation ("GMC") represents a second type of motion estimation that may efficiently encode image frames by estimating and compensating for global, frame-level image movement introduced by effects such as translation, zoom and angular rotation. Global motion estimation relies on local motion vectors derived in the course of local motion estimation and is therefore performed in conjunction with local motion estimation.

To compress an image frame, global motion compensation employs an affine model to estimate the nature, magnitude and direction of global motion within a frame. The global motion may be translation, zoom, rotation, or a combination of translation and zoom, translation and rotation, zoom and rotation, or translation, zoom and rotation. When a particular frame is encoded using global motion compensation, the encoded data comprises information regarding the trajectory of warping image points as measured between the encoded frame and one or more reference frames. Such motion information data is inserted in the MPEG-4 Video Object Plane header and transmitted in the bitstream.

Figure 6F illustrates an embodiment of global motion compensation that may be used in conjunction with motion estimation according to an aspect of the present invention. Figure 6F shows a global motion compensation system 672 comprising logic for encoding P, B, SB and other predicted frames in accordance with a global motion compensation format.

(a) Motion Vector Filtering

Global motion estimation seeks to identify frame-level motion of image that are associated with global movement of background or dominant object within the image. Localized movement variations within a frame that diverge from the global motion pattern may interfere with the global motion estimation process. Such localized variations may correspond to motion of individual objects within the frame. Elimination of such localized variations helps improve the accuracy and effectiveness of the global motion estimation process and may help isolate and remove localized motion within the image frame.

An aspect of the present invention provides a method for filtering local motion within an image frame by filtering motion vectors corresponding to pixels within the frame based on statistical attributes of the motion vectors. This method may identify and remove localized motion within the frame by comparing local statistical attributes of motion vectors within the frame with global statistical measures derived for motion vectors at a frame level.

In the embodiment of Figure 6F, a prefiltering module 674 performs local motion filtering based on statistical attributes of motion vectors. The prefiltering module 674 receives a set of motion vectors corresponding to individual macroblocks or blocks within a current frame being encoded. Some of these motion vectors may exhibit a certain amount of inconsistency with respect to the majority of the motion vectors within the frame, including, for example, variations in magnitude or direction. The prefiltering module 674 processes these motion vectors to selectively eliminate some of the motion vectors based on various criteria. In one case, the prefiltering module 674 eliminates motion vectors whose {x, y} components vary from a frame-level mean by more than a multiple of the corresponding frame-level standard deviation.

In the embodiment of Figure 6F, the prefiltering module 674 initially derives average values for the x and y components of all motion vectors within the frame based on the following formulas:

$$X_m = \frac{\sum_{i=1}^{N} x_i}{N} \text{ and}$$

$$Y_m = \frac{\sum_{i=1}^{N} y_i}{N},$$

where x_i and y_i are the components of a motion vector corresponding to a macroblock or block i in the frame, X_m and Y_m are the mean motion vector components determined over the whole frame, and N represents the number of motion vectors in the frame. The prefiltering module 674 then utilizes X_m and Y_m to determine additional statistical measures for the components of the motion vectors within the frame, including the standard deviation.

The prefiltering module 674 then processes each of the motion vectors corresponding to each macroblock in the frame and eliminates motion vectors whose x or y components deviate from the mean X_m and Y_m values by more than twice the corresponding standard deviation. In an alternative embodiment, the prefiltering module 674 may set the filtering threshold to three times the standard deviation.

(b) Affine Model Estimation and Motion Segmentation

The prefiltering module 674 serves as a preliminary filtering stage for motion vectors processed by an affine model estimation module 676. The affine estimation module 676 seeks to estimate an affine model that may be used to predict the motion vectors of a large number of pixels within the frame based on the location of particular pixels within the predicted frame. In an ideal case, if the affine model is perfectly accurate, a frame may be encoded based on a reference frame without storing any local motion vectors for individual macroblocks in the current frame. In practice, however, the affine model may not perfectly describe

the movement of all pixels within the current frame, and additional information may need to be transmitted to the decoder.

Figure 6G illustrates a flow chart of an affine model estimation and motion segmentation process that may be used in connection with global motion estimation according to an aspect of the present invention. In one embodiment, the affine model estimation and motion segmentation process 680 illustrated in Figure 6G may be accomplished by the affine estimation module 676 of the global motion estimation system 672 from Figure 6F.

The affine model estimation and motion segmentation process 680 comprises two phases: an affine model estimation phase and a motion segmentation phase. The processing performed at steps 682 and 684 helps determine the parameters for the affine model employed in global motion estimation according to this embodiment of the invention. Steps 686 and 688 implement a motion segmentation process that helps filter out local motion inconsistent with the global motion estimation model. The affine model estimation and the motion segmentation phases may be performed multiple times to progressively refine the affine model parameters.

As illustrated in Figure 6G, a set of motion vectors corresponding to macroblocks within a frame being encoded are received at step 682. The affine model estimation and motion segmentation process 680 of figure 6G utilizes these motion vectors as further described below to derive numerical values for the affine parameters.

Since the motion vectors received at step 682 may provide a mapping function between each macroblock in the current frame and a corresponding macroblock in a reference frame, and since the coordinates of macroblocks within the current frame may be readily determined, a set of affine equations connecting the current frame and the reference frame may be constructed. The number of these equations will usually exceed the number of affine model parameters (i.e., six). Consequently, various mathematical models may be

employed to derive values for the affine model parameters based on the large number of equations involving these parameters. In the embodiment of Figure 6G, least squares estimation is employed at step 682 to determine a set of initial values for the affine parameters. The affine model used in the present embodiment may be expressed as follows:

$$u_k = a_1 x_k + b_1 y_k + c_1, \text{ and}$$
$$v_k = a_2 x_k + b_2 y_k + c_2 \quad (5)$$

where u_k and v_k are the predicted x and, respectively y, components of a motion vector corresponding to a macroblock k in the current frame being encoded. The values of u_k and v_k can be determined independently of the affine model expressed in equation (5) based on motion vectors derived during local motion estimation. The variables x_k and y_k are the x, and respectively y, coordinates of the macroblock k within the frame. For example, the $\{x_0, y_0\}$ coordinates for the first macroblock in the frame, macroblock 0, are {0, 0}. The variables a_1, b_1, c_1, a_2, b_2 and c_2 are the affine model parameters being determined.

After initial values are derived for the parameters a_1, b_1, c_1, a_2, b_2 and c_2 at step 682, the affine model is employed at step 684 to construct a predicted frame on a pixel-by-pixel basis. To accomplish this, the macroblock coordinates $\{x_k, y_k\}$ of each macroblock in the current frame are sequentially substituted in the affine model to derive predicted values for u_k and v_k, denoted u'_k and v'_k:

$$u'_k = a_1 x_k + b_1 y_k + c_1, \text{ and}$$
$$v'_k = a_2 x_k + b_2 y_k + c_2. \quad (6)$$

Evaluation of equations (6) provides a motion vector corresponding to each macroblock in the current frame.

The motion vectors determined at step 684 are subsequently processed at steps 686 and 688 to eliminate inconsistent motion vectors that may be inconsistent with the affine model. This process corresponds to the motion segmentation phase of the affine model estimation 680. At step 686, the system selects a filtering threshold that defines a maximum allowed deviation from the corresponding motion vector derived during local motion estimation. In one implementation, the threshold has two components: a magnitude threshold and a phase threshold. The thresholds are pre-defined for each loop of the affine model estimation and motion segmentation process. A basic criterion for specifying the thresholds is that their values decrease after each loop.

At step 688, motion vectors whose attributes exceed the corresponding thresholds are eliminated as part of a filtering process. Generally, the lower the thresholds, the more motion vectors are eliminated during the motion segmentation phase. This filtering process is a part of motion segmentation and helps eliminate local motion inconsistent with the global motion model.

The system then proceeds to step 690 where a decision is made whether the performance of the affine model using the current values of the affine parameters provides sufficient accuracy to encode the current frame. If the system determines that encoding the frame based on the current affine parameters would not be sufficiently efficient or accurate, the system returns to step 682 and repeats the affine model estimation and the motion segmentation processing described above.

While the processing during subsequent passes through steps 682, 684, 686 and 688 is substantially as described above, certain differences may exist. For example, macroblocks whose motion vectors were eliminated through filtering at step 688 may be excluded from

further consideration. Further, during subsequent passes, the values of the filtering thresholds defined at step 686 may be decreased to further improve the accuracy of the global motion estimation process. In a particular implementation, during the fifth loop through the motion segmentation phase, the filtering thresholds are set to 0.5 pixels for the magnitude of the motion vectors and 5 degrees for the phase.

In one embodiment, the system ceases refinement of the affine model parameters after concluding a certain number of loops. In one particular implementation, the system exits the affine model estimation and motion segmentation process 680 after five loops. If the maximum number of loops is reached, or if the accuracy of the affine model is adequate, the system exists the affine model estimation and motion segmentation process 680 at step 690

The evaluation at step 692 produces the final affine model. Since some of the motion vectors corresponding to pixels in the current frame were eliminated during the first or subsequent passes through step 688, only certain pixels in the current frame retain motion vectors. The pixels that still have motion vectors in the current frame upon execution of step 692 are thereafter considered to be part of the global picture that conforms to, and may be predicted from, the affine model. Upon transmission of the affine model coefficients to the decoder together with other information, the predicted, global motion compensated version of the current frame, may be reconstructed.

(c) Warping

In addition to the motion vectors determined during affine model estimation and motion segmentation for selective motion vectors in the current frame, information regarding the luminance and chrominance of the respective pixels may also need to be encoded and transmitted to the decoder. Luminance and chrominance data corresponding to each macroblock is organized in 8x8 blocks, as previously described in connection with the embodiment of Figure 6A.

Luminance values for pixels normally vary from 0 to 255 and may require up to eight bits to be encoded. One technique for reducing the number of bits necessary for encoding luminance values for the pixels included in the global motion compensated image is to encode differential luminance values. This technique may be particularly effective in this case because the luminance values of pixels in the current frame and their corresponding pixels in the reference frame tend to be close in magnitude. Differential encoding of luminance values may be accomplished in an embodiment of the invention using a warping module 678 comprised in the global motion estimation system 672.

When the system exits the affine model estimation and motion segmentation process 680, the motion vectors produced at step 692 of the embodiment of Figure 6G are transmitted to the warping module 678 from the embodiment of Figure 6F. The warping module 678 processes the current frame, the motion vectors derived based on the affine model, and the corresponding reference frame to determine pixel-by-pixel residual luminance values for pixels in the current frame. To accomplish this, the warping module 678 subtracts the luminance value of each pixel in the current frame that retains a motion vector from the luminance value of the corresponding pixel in the reference frame identified by the corresponding motion vector. Alternatively stated, for each pixel that retains a motion vector in the current frame, the warping module 678 identifies the corresponding reference pixel in the reference frame and subtracts the luminance values of the two pixels to obtain a differential value, denoted a residual value.

This selective pixel-by-pixel subtraction operation produces a residual luminance value corresponding to each pixel processed in the current frame. These residual values may be encoded for each respective pixel and transmitted to the decoder together with the corresponding motion vectors to reconstruct the original frame. Since the luminance values of pixels in the current frame and their corresponding pixels in the reference frame tend to be close in

magnitude as a result of the global motion estimation process, the residual luminance values tend to be close to zero and therefore may be encoded with less bits. This helps decrease the encoded data rate.

B. Coding Decision

In the system for compressing video illustrated in Figure 4, the motion vectors derived by the motion estimation system 504 based on local motion estimation and global motion estimation are then transmitted to the coding decision system 506 together with additional information that may be necessary to encode the video signal, as previously described. The coding decision system 506 utilizes these motion vectors to compress and encode each individual frame at macroblock or block level based on a preferred encoding method for each macroblock. For each macroblock being encoded, the coding decision system 506 produces corresponding encoding data, which comprises a prediction error, one or more reference motion vectors, and header and other overhead information. The coding decision system 506 transmits such encoding data to the rate distortion optimizer 510 for further processing.

The coding decision system 506 comprises a mode processing subsystem 508, a rate distortion optimization subsystem 510 and an encoding selection subsystem 512. The mode processing subsystem 508 applies various encoding modes to each macroblock of each frame being encoded to produce a set of candidate encoding formats. The rate distortion optimization subsystem 510 processes each of the candidate encoding formats for each macroblock to determine a set of corresponding cost functions. The encoding selection subsystem 512 compares the cost functions to identify a preferred encoding format for each macroblock. Data compressed in accordance to the preferred encoding format is then transmitted by the coding decision system 506 as an output encoded bistream.

The following discussion describes the operation of the coding decision system 506 in further detail.

i. Multiple Reference Frame Selection

An aspect of the invention provides a multiple reference frame selection method for encoding a macroblock in accordance to various encoding modes based on reference macroblocks comprised in at least two reference frames. By selecting reference macroblocks from more than one reference frame, this aspect of the invention expands the scope of the search for reference macroblocks and may identify reference macroblocks better suited to serve as a basis for encoding the current macroblock. This may help increase the compression of the underlying video data and the quality of the encoding process.

Multiple reference frame selection may be utilized in connection with various encoding modes, including multiple hypothesis, overlapped motion compensation, global motion compensation, global motion compensation forward, global motion compensation backward, and direct. These and other encoding modes are further described in connection with the coding decision system 700 illustrated in Figures 7A and 7B. Multiple reference frame selection may also be employed to provide an expanded set of reference macroblocks or blocks for motion vector determination based on local motion estimation. In a particular case, multiple reference frame selection is employed to encode P, B, SB and SSB frames.

Multiple reference frame selection may be accomplished by logic that receives and processes a set of candidate reference frames to select a subset of reference frames suitable to serve as a basis for encoding other frames. In one embodiment illustrated in Figure 7C and further described below, multiple reference frame selection is implemented via a reference frame selection system 758 that produces reference frames to be used by a multiple hypothesis system 740 as a

basis for encoding macroblocks according to a multiple hypothesis format.

The number of reference frames produced by multiple reference frame selection may depend on various factors, including, for example, the nature of the underlying video sequence, characteristics of the corresponding application, or size of the storage media available at the encoder or decoder. In one case, if the video signal being encoded comprises numerous scene changes, multiple reference frame selection may produce fewer reference frames as the probability of identifying suitable reference macroblocks in reference frames further away from the current frame decreases. In another case, if the memory available for buffering reference frames at the encoder or at the decoder is low, the number of reference frames produced by multiple reference frame selection may be correspondingly limited.

The reference frames produced by multiple reference frame selection are stored in a reference frame buffer, possibly to serve as a basis for encoding a current macroblock. The reference frame buffer may be comprised within a dedicated multiple reference frame selection system such as the reference frame selection system 758 from Figure 7C, or may be included in an encoding module such as the multiple hypothesis system 740.

The reference frame buffer used in connection with multiple reference frame selection may store various types of frames, depending on the format of the current frame being decoded. For example, in the embodiment of Figure 7A where the coding decision system 700 encodes macroblocks or blocks comprised in a current P frame, the reference buffer stores I or P frames that may serve as a basis for encoding the current P frame. In an alternative embodiment, if the current frame is an SB frame for example, the reference buffer may also include B frames since SB frames may be encoded based on I, P and/or B frames.

In one implementation, frames are stored in the reference buffer in temporal order, i.e., the most recently processed reference picture is the first frame in the buffer, at position 1. Once the current I or P frame is processed, a queue operation is implemented to prepare the buffer for the processing of the next P frame. The previously-processed reference frame is pushed to the front of the queue (i.e., at position 1), and all other reference frames in the queue are moved up one position through a shift operation, in the natural temporal direction. The formerly last frame in the queue is discarded. This mechanism is implemented sequentially, for all the frames in the frame sequence.

In one implementation, if the current frame being decoded is an I frame, the reference buffer may be reset when the processing for that frame is completed. The buffer may have a variable size. Consider, for example, an exemplary frame sequence comprising I, P and B frames organized according to a pattern I B B P B B P B B P B B P B B P B B I B B P B B P B B P. For a particular value of M = 4, the first P picture may have one reference frame (I), the second P picture may have two reference frames (I, P), and so forth, with the buffer increasing to a maximum size of 4. As soon as the second I picture is processed, the buffer may be reset, depending on the nature of the corresponding application. Therefore, the first P picture following the second I picture may again only have one reference frame (I).

Certain aspects of the present invention provide rules for determining motion vectors for each macroblock in the current frame when the motion vectors point to more than one reference frame in the reference buffer. Commonly, each macroblock motion vector is encoded based on a set of neighboring predictor motion vectors, denoted as "motion vector context" or "predictor motion vector candidates." In one embodiment, if all the candidate predictors point to the same reference frame as the current motion vector being encoded, the normal baseline MPEG-4 rules are applied to select the x and y components of the current motion vector. If only two of the three motion vectors from the

context point to the same reference frame as the current motion vector, the x and y components of the third motion vector candidate are set to zero and the current motion vector is encoded as the median of the three predictor candidates. If only one out of the three predictor candidates points to the same reference frame as the current motion vector to be encoded, that vector is selected as the predictor. If none of the predictors point to the same reference frame as the current motion vector, the predictor for the current motion vector is set to {0, 0}.

In one embodiment, the rules for determining motion vectors for macroblocks in a current frame described above apply only when the motion vector context includes a set of three predictor motion vector candidates. Under certain circumstances, however, macroblocks may be encoded using less than three predictor motion vector candidates. This may happen, for example, when macroblocks are located along the boundary of the image. Under such circumstances, motion vectors may be encoded in accordance with the MPEG-4 standard.

To process data encoded using multiple reference frame selection, additional information is introduced in the bitstream and transmitted to the decoder. Compared to overhead data normally transmitted in common MPEG implementations, this information may include additional bits inserted in the headers at various description levels (e.g., in the Video Object Layer header or in the macroblock header). Various relevant variables utilized in a particular implementation of the present invention will be described below.

ii. Mode Processing

To encode a frame in accordance with various aspects of the present invention, the frame is partitioned into macroblocks of variable sizes, and the macroblocks are encoded on an individual basis. Encoding macroblocks of a frame on an individual basis instead of encoding the complete frame in a single process provides a number of advantages, including the flexibility of selecting different encoding

modes for different macroblocks within the frame. The ability to encode different macroblocks comprised in a single frame based on different models permits exploitation of image attributes that may exist at a local scale within the image, but that may not apply to the complete frame. For example, certain objects within a particular frame may exhibit significant visual similarities with corresponding objects in preceding or subsequent reference frames, while the particular frame as a whole may not. Processing that particular frame at a macroblock level may permit isolation of those objects within macroblocks and encoding of the objects in reference to the corresponding objects in one or more reference frames.

The mode processing subsystem 508 applies various encoding modes to each macroblock of each frame being encoded to produce a set of candidate encoding formats. Depending on the type of frame being encoded (i.e., P, B, SB or SSB frame), different encoding modes may be employed, as further described below.

(a) P Frame Macroblock Encoding Modes

Encoding of P frame macroblocks differs from encoding B, SB or SSB frame macroblocks in a number of aspects, including the fact that P frame macroblocks may only be predicted in a forward direction, based on a preceding frame, while B, SB and SSB frame macroblocks may be predicted both in a forward and a backward direction, based on preceding and/or subsequent frames. As a result of this difference, an aspect of the invention utilizes different modes to encode P frame macroblocks as compared to B, SB and SSB frame macroblocks.

Figure 7A illustrates a system for encoding a P frame based on selecting an encoding mode from a plurality of encoding modes according to an embodiment of the present invention. The coding decision system 700 from the embodiment of Figure 7A comprises a P frame mode processor 704, a rate distortion optimizer 706 and a

selector 708. The P frame mode processor 704 applies sequentially various encoding modes to a P frame macroblock being encoded.

For each encoding mode, the P frame mode processor 704 transmits the encoded macroblock to the rate distortion optimizer 706, where the encoded macroblock is converted into a corresponding bitstream suitable for transmission over a communication channel or storage on a storage medium. The bitstream data corresponding to each mode is then processed by the selection module 708, which employs a decision module 722 to select from the candidate encoding modes a preferred encoding mode that provides the most efficient macroblock encoding scheme. This decision relies on a cost function derived by the rate distortion optimizer 706 for each mode.

The P frame mode processor 704 comprises a number of systems that may encode a P frame macroblock based on the following encoding modes: intra, inter, inter 4V, multiple hypothesis, overlapped motion compensation and global motion compensation. Each of these encoding modes provides a particular format for encoding a macroblock, with a corresponding cost function. The systems comprised in the P frame mode processor 704 that implement these encoding modes are further described below together with their corresponding encoding modes.

(1) Intra Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock based on the intra encoding mode is performed substantially as taught in MPEG-4. Generally, to encode a macroblock according to the intra encoding mode, the luminance and chrominance data corresponding to the macroblock is encoded without reference to another macroblock or frame. A macroblock encoded based on the intra encoding mode is generally self-contained and may be decoded independently of other macroblocks or frames. An intra-encoded macroblock has no motion vector.

As illustrated in the embodiment of Figure 7A, an intra mode module 710 comprised in the P frame mode processor 704 receives data corresponding to a current macroblock and encodes that data in accordance with the P frame intra encoding mode. The intra mode module 710 comprises logic for processing luminance, chrominance and other data associated with the current macroblock, and for encoding this data according to the intra mode format. Upon encoding the data corresponding to the current macroblock, the intra mode module 710 transmits this data to the rate distortion optimizer 706 for further processing.

(2) Inter Macroblock Encoding Mode

Unlike the intra encoding mode, the inter encoding mode generally utilizes motion vectors to encode a macroblock based on information obtained from a reference macroblock and/or a reference frame. One embodiment of the invention utilizes a conventional MPEG-4 inter encoding method to encode macroblocks according to the inter macroblock encoding mode.

An inter mode module 712 comprised in the P frame mode processor 704 receives data corresponding to a current macroblock. This data comprises luminance and chrominance information for each pixel in the current macroblock. The luminance and chrominance data is organized in 8x8 blocks, as previously described in connection with the embodiment of Figure 6A. In addition to luminance and chrominance data, the inter mode module 712 also receives a set of motion vectors from a local motion estimation system 702. The local motion estimation system 702 may be the motion estimation system 504 from the embodiment of Figure 5. These motion vectors identify a particular reference macroblock within a corresponding reference frame.

The inter mode module 712 processes the luminance and chrominance data together with the motion vectors to encode the current macroblock based on a reference macroblock. To minimize the number

of bits used to encode the current macroblock, the inter mode module 712 takes advantage of the temporal redundancies present in the image data. To achieve that, the inter mode module 712 subtracts luminance and chrominance values for each pixel in the current frame from luminance and chrominance values of corresponding pixels in the reference macroblock to obtain differential luminance and chrominance values for each pixel in the current frame. The differential luminance and chrominance values are denoted as "residual" values or "prediction error."

Since the luminance and chrominance values for each pixel in the current macroblock tend to be similar to the luminance and chrominance values of the corresponding pixel in the reference macroblock, the residual luminance and chrominance values for each pixel tend to be small. Consequently, these residual luminance and chrominance values may be encoded using a lower number of bits, which may help decrease the data rate.

Pixels in the reference frame corresponding to pixels in the current frame are identified based on the motion vectors received by the inter mode module 712 from the local motion estimation system 702. Since in one embodiment of the invention P frames are encoded based on preceding reference frames in the frame sequence, the motion vectors employed to encode P frame macroblocks point in a forward direction. These motion vectors are denoted as "forward motion vectors."

The inter mode module 712 processes the luminance and chrominance residual values and the forward motion vectors associated with the current macroblock and encodes them. The inter mode module 712 then transmits this encoded data to the rate distortion optimizer 706 for further processing.

(3) Inter 4V Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock based on the inter 4V encoding mode is performed substantially as taught in MPEG-4.

An inter 4V module 714 comprised in the P frame mode processor 704 receives data corresponding to the current macroblock. This data comprises luminance and chrominance information for each pixel in the current macroblock. The inter 4V module 714 also receives from the local motion estimation system 702 multiple forward motion vectors that identify corresponding reference blocks in a particular reference frame. The inter 4V module 714 further receives chrominance and luminance data for pixels in the reference macroblock.

The inter 4V module 714 processes macroblocks at a sub-macroblock level by partitioning the macroblocks into smaller blocks and encoding these blocks individually based on corresponding reference blocks comprised in various reference frames. In some implementations, the blocks may have rectangular and/or square shapes. The size and number of blocks may also vary: as the size of blocks decreases, the number of blocks per macroblock increases. Decreasing the size of the blocks may provide better encoding accuracy and increased compression.

In one embodiment, the inter 4V module 714 partitions each 16x16 macroblock into four 8x8 blocks, denoted blocks, and receives from the local motion estimation system one or more forward motion vectors that identify corresponding reference blocks in a particular reference frame.

The luminance and chrominance data for both the current and the reference macroblocks is organized in 8x8 blocks, as previously described in connection with the embodiment of Figure 6A. The luminance and chrominance data may correspond to blocks processed using fractional pixel refinement. The inter 4V module 714 processes luminance and chrominance blocks by subtracting corresponding pixel values associated with the current and reference blocks to produce

residual luminance and chrominance values substantially as previously described in connection with the inter module 712. In one embodiment of the invention, chrominance blocks associated with the current macroblock are predicted based on the motion vectors corresponding to the luminance blocks.

The inter 4V module 714 then encodes the residual luminance and chrominance values for pixels in the current macroblock together with the forward motion vectors corresponding to the current macroblock and other relevant information and transmits these values to the rate distortion optimizer 706 for further processing.

(4) Multiple Hypothesis Macroblock Encoding Mode

Multiple hypothesis is a macroblock encoding mode that provides a method for encoding a macroblock of a current frame based on combinations of two or more reference macroblocks. In an embodiment of the present invention, macroblock encoding in accordance with the multiple hypothesis format is accomplished by the multiple hypothesis module 716 comprised in the P frame mode processor 704 from Figure 7A.

As illustrated in Figure 7A, the multiple hypothesis module 716 receives a current macroblock to be encoded according to the multiple hypothesis format. The multiple hypothesis module 716 also receives a set of reference frames comprising reference macroblocks that may serve as a basis for encoding the current macroblock. The multiple hypothesis module 716 further receives a corresponding set of motion vectors from the local motion estimation system 702. The motion vectors identify particular reference macroblocks within the reference frames corresponding to the current macroblock. The multiple hypothesis module 716 encodes the current macroblock based on linear combinations of reference macroblocks and determines a set of corresponding prediction errors that may be utilized to reconstruct the

current macroblock. The encoding data corresponding to the use of linear combinations of reference macroblocks is transmitted to the rate distortion optimizer. The rate distortion optimizer 706 processes the encoding data, which comprises a corresponding prediction error, motion vectors, and overhead information for each combination of reference macroblocks, to evaluate the desirability of encoding the current macroblock based on each particular combination.

Figure 7C illustrates a system for encoding a macroblock using a multiple hypothesis macroblock encoding mode according to an embodiment of the present invention. In one implementation, the method illustrated in Figure 7C may be executed by logic within the multiple hypothesis module 716 comprised in the P frame mode processor 704 from Figure 7A.

As illustrated in Figure 7C, a reference frame selection system 758 identifies a set of reference frames comprising reference macroblocks that may serve as a basis for encoding the current macroblock based on the multiple hypothesis mode. A method for multiple reference frame selection provided by an aspect of the present invention was previously described. Multiple reference frame selection generally provides a method for utilizing an expanded set of reference frames and reference macroblocks as a basis for encoding a present macroblock.

As illustrated in Figure 7C, the reference frame selection system 758 produces a set of reference frames and transmits these reference frames to the multiple hypothesis system 753. In addition to the reference frames received from the reference frame selection system 758, the multiple hypothesis system 753 receives a set of motion vectors from the local motion estimation system 757. The local motion estimation system 757 may be the local motion estimation system 702 from Figure 7A and/or the local motion estimation system 670 from Figure 6F.

The motion vectors received by the multiple hypothesis system 753 identify particular reference macroblocks within the reference frames that exhibit a relatively-high degree of similarity with the current macroblock. These reference macroblocks may be the closest representations of the current macroblock that the local motion estimation system 757 identified within the reference frames. In one case, each of the motion vectors transmitted by the local motion estimation system 757 to the multiple hypothesis system 753 identifies a best-match reference macroblock within a corresponding reference frame, for the current macroblock.

The multiple hypothesis system 753 receives a total of N reference macroblocks. In one case, the multiple hypothesis system 753 receives a set of M reference frames, and each reference frame comprises a number P of reference macroblocks. In this case, therefore, the total number of reference macroblocks received by the multiple hypothesis system 753 is N = M*P.

In one embodiment, each of the N reference macroblock corresponds to a reference motion vector received from the local motion estimation system 757. In an alternative embodiment, some or all of the N reference macroblocks are determined based on a comprehensive search through reference frames. In this alternative embodiment, instead of relying on reference macroblocks identified by local motion estimation as best matches with respect to the current macroblock, the multiple hypothesis system 753 searches for better suited reference macroblocks within the reference frames.

To identify reference macroblocks that may be better suited to serve as a basis for encoding the current macroblock, the multiple hypothesis system 753 sequentially partitions each reference frame into reference macroblocks of an appropriate size (e.g., 16x16 pixels). In one case, the multiple hypothesis system 753 accomplishes this by identifying the top left macroblock in the reference frame and then progressively and exhaustively stepping through the reference frame

one pixel at the time, horizontally and vertically, to identify all other possible macroblocks within the frame. Alternatively, the multiple hypothesis system 753 may limit its search to an arbitrary area centered around a best-match reference macroblock identified during local motion estimation. Generally, as the size of the search area increases, the number of reference macroblocks identified by the macroblock search module 752 and the processing complexity also increase.

In a particular implementation, the multiple hypothesis system 753 receives a set of motion vectors from the local motion estimation 746, where each motion vector identifies a particular reference macroblock within a corresponding reference frame. The multiple hypothesis system 753 then utilizes these particular reference macroblocks as a basis for further processing. There may be more than one motion vector associated with some or all of the reference frames. In a particular case, the local motion estimation system 757 transmits five motion vectors for each reference frame provided by the reference frame selection system 758, wherein the five motion vectors correspond to the best five candidate reference macroblocks identified during local motion estimation. In this case, the reference frame selection system 758 relies on five reference macroblocks corresponding to each reference frame to encode the current macroblock based on the multiple hypothesis format.

As illustrated in Figure 7C, a selector module 754 groups the reference macroblocks received by the multiple hypothesis system 753 in various formations to produce groups of reference macroblocks that may serve as a basis for encoding the current macroblock. In one case, the combination module 754 combines the N reference macroblocks into all possible unique groups of K reference macroblocks. The total number of such groups would be:

$$\frac{N!}{K!(N-K)!} = \frac{(MP)!}{K!(MP-K)!}.$$

For example, if the selector module 754 selects groups of K=3 reference macroblocks out of a total of N=5 reference macroblocks, the selector module 754 may produce up to 10 combinations of reference macroblocks. The selector module 754 may retain all the combinations of reference macroblocks produced, or may selectively eliminate some of the combinations.

In a different implementation, the selector module 754 may instead produce a number of pairs of reference macroblocks that may serve as a basis for predicting the current macroblock. In one implementation, the selector module 754 produces all possible pairs of reference macroblocks. In this case, for N reference macroblocks, the selector module 754 produces a total of

$$\frac{N!}{2!(N-2)!} = \frac{(MP)!}{2!(MP-2)!}$$

pairs of reference macroblocks. For example, for N=4 reference macroblocks, the selector module 754 would produce a total of 6 pairs (K=2) of reference macroblocks. In an alternative implementation, the selector module 754 may selectively eliminate some of these pairs and may retain only a subset of the total number of pairs.

The selector module 754 transmits the combinations of reference macroblocks produced and retained as described above to a predictor module 755. The predictor module 755 also receives a set of weights. The predictor module 755 combines the weights with the reference macroblocks received from the selector module 754 to produce a set of pixel-by-pixel linear combinations of reference macroblocks. The linear combinations may apply to various pixel attributes, including luminance or chrominance.

In one implementation, a representation of the current macroblock based on $K \geq 2$ reference macroblocks may be determined on a pixel-by-

pixel basis by applying the following formula to luminance values of corresponding pixels:

$$PMB = \sum_{i=1}^{K} \mathrm{W_i} * \mathrm{RMB_i} + C \,. \qquad (7)$$

In equation (7), PMB represents the combined predictor macroblock corresponding to the current macroblock. K, the upper limit of the summation, represents the number of reference macroblocks that are combined to produce PMB for each combination of macroblocks. RMB_i represent the individual reference macroblocks selected to predict the current macroblock in the current combination. W_i represent matrix weights that weigh the reference macroblocks on a pixel-by-pixel basis. In one embodiment, the values of the weights W_i are predetermined and are stored at the decoder. In another embodiment, the values of W_i are dynamically determined based on the nature of the current and/or reference frames and are transmitted to the decoder. The values of the weights may be the same for the whole macroblock, or may vary on a pixel-by-pixel basis. C represents a pixel-level matrix constant. In a particular implementation, the values of W_i and C are ½ for each pixel.

The predictor module 755 processes according to equation (7) each of the combinations of reference macroblocks transmitted by the selector module 754 to produce corresponding pixel-by-pixel predicted macroblocks. For each of these predicted macroblocks, an error predictor module 756 determines a corresponding prediction error that evaluates luminance and/or chrominance differences between the current macroblock and the predicted macroblock on a pixel-by-pixel basis. The prediction error may be determined at a pixel level using the formula

$$PE = MB - PMB, \qquad (8)$$

where PE represents the prediction error for the current macroblock, PMB represents the combined predictor macroblock determined based on equation (7), and MB represents the current macroblock being encoded.

For each of the combined predictor macroblocks, the multiple hypothesis system 753 then transmits to a rate distortion optimizer system 760 the corresponding motion vectors received from the local motion estimation system 757 and the corresponding prediction error determined by the error predictor module 756. In one case, the rate distortion optimizer 750 is the rate distortion optimizer 706 from the embodiment of Figure 7A. The rate distortion optimizer 750 processes the encoding data transmitted by the multiple hypothesis system 753 corresponding to each of the combined predictor macroblocks to select a particular combination of reference macroblocks that provides the most efficient encoding basis for the current macroblock.

(5) Decoding of Macroblocks Encoded According to the Multiple Hypothesis Mode

To reconstruct a frame encoded in accordance with the multiple hypothesis format, the decoder uses a process which is substantially the reverse of the encoding process. Decoding of the frame proceeds at a macroblock level, with individual macroblocks being individually decoded and inserted in the proper position within the frame. To decode a particular multiple hypothesis encoded macroblock, the decoder first identifies the set of reference frames that comprise the corresponding predictor macroblocks RMB_i used to encode that particular macroblock. Prior to decoding the current macroblock, the corresponding reference frames have already been decoded. The corresponding reference frame may be stored in a decoder reference frame buffer.

The decoder than extracts from the bitstream the motion vectors that identify the position of the corresponding predictor macroblocks RMB_i within the corresponding reference frames. The decoder then

extracts the predictor macroblocks RMB_i and combines them with the weights W_i and the constant C in accordance with equation (7) to reconstruct the corresponding combined predictor macroblock PMB. The decoder then extracts from the bitstream the prediction error PE corresponding to the combined predictor macroblock PMB and adds it to the combined predictor macroblock PMB on a pixel-by-pixel basis to reconstruct the original macroblock.

In one implementation, encoding and decoding of frames using the multiple hypothesis format employs a number of additional fields in the conventional MPEG-4 syntax. Various relevant variables utilized in a particular implementation of the present invention will be described below.

(6) Overlapped Block Motion Compensation Macroblock Encoding Mode

Overlapped block motion compensation (OBMC) is a macroblock encoding mode that provides a method for encoding a current macroblock by predicting individual pixels within the macroblock based on reference data comprised in a set of reference frames. To encode a macroblock according to the overlapped motion compensation format, the macroblock is partitioned into smaller blocks, and each block is encoded on a pixel-by-pixel basis. Overlapped motion compensation helps reduce noise and blocking artifacts by, among others, providing smoother transitions across block boundaries. In an embodiment of the present invention, macroblock encoding in accordance with the overlapped motion compensation format is accomplished by the overlapped motion compensation module 718 comprised in the P frame mode processor 704 from Figure 7A.

As illustrated in Figure 7A, the overlapped motion compensation module 718 receives a current macroblock to be encoded according to the OBMC format. The overlapped motion compensation module 718 also receives a set of reference frames that may serve as a basis for

encoding the current macroblock. The overlapped motion compensation module 718 further receives a set of reference motion vectors from the local motion estimation system 702. The overlapped motion compensation module 718 encodes the current macroblock based on linear combinations of reference pixel predictors and determines a corresponding prediction error that may be utilized to reconstruct the current macroblock. Encoding data comprising the corresponding prediction error, motion vectors and other overhead information is transmitted to the rate distortion optimizer 706 for further processing. The rate distortion optimizer 706 processes this encoding data to evaluate the desirability of encoding the current macroblock or block according to the overlapped motion compensation format.

Figure 7D illustrates a flow diagram for encoding a macroblock using an OBMC encoding mode according to an embodiment of the present invention. The overlapped motion compensation encoding process 761 illustrated in Figure 7D may be performed by logic within the OBMC encoding module 718 comprised in the P frame mode processor 704 from Figure 7A.

Overlapped motion compensation encoding may be performed with respect to various pixel attributes, including, for example, luminance and chrominance. In one implementation, the overlapped motion compensation process 761 performs overlapped motion compensation only for the luminance component (Y) of the pixels in the macroblock being encoded. In alternative implementations, overlapped motion compensation also includes pixel chrominance.

An aspect of the invention provides a method for encoding a macroblock in accordance with an overlapped motion compensation encoding mode based on reference macroblocks comprised in at least two reference frames. In one case, the reference frames are determined using multiple reference frame selection. Multiple reference frame selection was previously described. In one embodiment, a reference frame selection system 769 comprises logic for processing candidate

reference frames based on multiple reference frame selection. The structure and operation of the reference frame selection system 769 may be substantially identical with the structure and operation of the reference frame selection system 758 from Figure 7C. By selecting reference macroblocks from more than one reference frame, this aspect of the invention expands the scope of the search for reference macroblocks and may identify reference macroblocks better suited to serve as a basis for encoding the current macroblock. This may help increase the compression of the underlying video data and the quality of the encoding process.

The overlapped motion compensation process 761 receives a current macroblock to be encoded at step 763 and partitions this macroblock into four blocks. The blocks are substantially identical to each other in size and contain substantially the same number of pixels. In a particular case, each 16x16 macroblock is partitioned into four 8x8 blocks. In an alternative implementation, the blocks may have different relative sizes.

The OBMC encoding process 761 processes the blocks comprised in the current macroblock sequentially. At step 765, the OBMC encoding process 761 selects a particular block within the current macroblock to be encoded. At step 766, the OBMC encoding process 761 selects a particular pixel within the current block for pixel-level processing. In subsequent passes, the remaining pixels in the current block will be sequentially selected at step 766 and processed analogously to the current pixel.

The OBMC encoding process 761 receives at step 770 a set of reference frames from the reference frame selection system 769. The OBMC encoding process 761 may alternatively rely solely on reference pixels comprised in a single reference frame, in which case the encoding process may conform to conventional MPEG-4 overlapped motion compensation encoding syntax.

In addition to reference frames, the overlapped motion compensation encoding process 761 also receives a set of motion vectors identifying particular blocks within the reference frames at step 770. Each of these motion vectors corresponds to a block within the current frame and points to a reference block within a corresponding reference frame. Each of these reference blocks was identified during local motion estimation processing to exhibit a relatively-high degree of similarity with a corresponding block in the current frame.

The overlapped motion compensation encoding process 761 processes the motion vectors and the reference frames at step 770 to identify corresponding predictor pixels for the current pixel selected at step 766. To accomplish this, the system locates within the current frame a set of neighboring blocks disposed proximally to the current block selected at step 765. The neighboring blocks of the current block in the current frame are denoted "context blocks." In a particular embodiment, the context of the current block comprises two blocks that share a common side with the current block.

Figure 7E illustrates a context that may serve as a basis for encoding a current block based on an overlapped motion compensation encoding mode, according to an embodiment of the present invention. As illustrated in Figure 7E, a current 16x16 macroblock 792 is partitioned into four 8x8 blocks, including a current block 792. The current block 792 is partitioned into four 4x4 quadrants, including a current quadrant 794. The context of the current block 792 consists of a first context block 796 and a second context block 798. The selection of the current context blocks depends on the position of the current pixel 761 within the current block being processed.

The system processes the motion vectors received from the local motion estimation 768 to identify motion vectors corresponding to the current block and to the context blocks. In the embodiment of Figure 7F, the system retrieves a motion vector A corresponding to the first context block 798, a motion vector B corresponding to the second context block

796, and a motion vector C corresponding to the current block 792. Each of these motion vectors identifies a particular predictor pixel in a corresponding reference frame. The selection of the context blocks and thus of the motion vectors depends on the quadrant in which the current pixel 791 is situated in the current block.

The system translates each of the motion vectors A, B and C into the coordinate position of the current pixel 791 to obtain a set of corresponding translated motion vectors A', B' and C'. Translation of vectors may be achieved by modifying the {x, y} components of the vectors. Each of the translated motion vectors A', B' and C' now points to a particular predictor pixel in a corresponding reference frame. These predictor pixels provide a basis for encoding the current pixel 791.

To encode the current pixel, the predictor pixels are weighted by a set of weights that vary based on the relative position of the current pixel within the current block. These weights are predefined at both the encoder and the decoder and are retrieved at step 773. In one implementation, determination of a motion vector for the pixel under consideration employs the weight matrices and process commonly used in MPEG-4. Alternatively, these weights may be determined dynamically at step 773, in which case the weights would be encoded and transmitted to the decoder in the bitstream together with the rest of the data.

At step 774, the current pixel 791 is encoded based on a linear combination of the corresponding predictor pixels to produce a corresponding combined predictor pixel. The linear combination of predictor pixels operates on a particular pixel attribute, such as luminance or chrominance. To encode the luminance of the current pixel, for example, luminance values of the predictor pixels are combined to produce a combined predictor luminance value for the current pixel.

At step 774, the overlapped motion compensation process 761 evaluates the difference between the luminance values of the current

pixel and of the combined predictor pixel to produce a residual luminance value corresponding to the current pixel. The residual luminance value represents a prediction error that may be utilized to recover the current pixel from the combined predictor pixel.

Determination of residual values, prediction errors and reconstruction of original macroblocks was previously discussed in connection with the global motion estimation system 672 from Figure 6F and the multiple hypothesis system 753 from Figure 7C.

At step 775, the overlapped motion compensation process 761 determines whether all pixels in the current block have been processed. If additional pixels remain unprocessed, the overlapped motion compensation process 761 returns to step 766, where it selects the next pixel in the current block to be processed. If all pixels in the current block have been processed, the overlapped motion compensation process 761 progresses to step 776. At step 776, a determination is made whether all blocks in the current macroblock have been processed. If additional blocks remain unprocessed, the overlapped motion compensation process 761 returns to step 765, where it selects the next block.

If all pixels and blocks in the current macroblocks have been processed, the overlapped motion compensation process 761 transmits to the rate distortion optimizer 762 the corresponding motion vectors, the prediction errors derived for each pixel in the current macroblock and other overhead data for further processing. The distortion optimizer 762 processes the data provided by the overlapped motion compensation process 761 to assess the desirability of encoding the current macroblock based on the overlapped motion compensation encoding mode.

One implementation of the overlapped motion compensation format provided by aspects of the present invention is accomplished through a number of extensions to the standard MPEG-4 syntax.

Various relevant variables utilized in a particular implementation of the present invention will be described below.

For each macroblock being decoded, the decoder determines whether the overlapped motion compensation mode was employed to encode the macroblock.

Decoding of a macroblock encoded using the overlapped motion compensation format employs a process substantially opposite to the encoding process described above.

(7) Global Motion Compensation Encoding Mode

Global motion compensation represents an encoding mode that may efficiently encode image frames by estimating and compensating for global, frame-level image movement introduced by effects such as translation, zoom and angular rotation. Global motion estimation normally relies on motion vectors derived in the course of local motion estimation and is therefore normally performed in conjunction with local motion estimation.

To compress an image frame, global motion compensation employs an affine model to estimate the nature, magnitude and direction of global motion within a frame. The global motion may be translation, zoom, rotation, or a combination of translation and zoom, translation and rotation, zoom and rotation, or translation, zoom and rotation. When a particular frame is encoded using global motion compensation, the encoded data comprises information regarding the trajectory of warping image points as measured between the encoded frame and one or more reference frames. Such motion information data is inserted in the MPEG-4 Video Object Plane header and transmitted in the bitstream.

As illustrated in Figure 7A, a global motion compensation module 720 comprised in the P frame mode processor 704 receives a set of motion vectors from the local motion estimation system 702 together with a set of reference frames and a current frame being encoded. The

global motion compensation module 720 processes this data to filter out local motion artifacts identified in the current frame with respect to the reference frames and encodes the current macroblock based on one or more reference frames. The global motion compensation module 720 then transmits the data corresponding to the encoded representation of the current frame to the rate distortion optimizer 706 for further processing.

The structure and operation of a global motion compensation system provided by various aspects of the present invention was previously described in connection with the global motion estimation module 604 comprised in the motion estimation system 600 from Figure 6A and the global motion estimation system 672 from Figure 6F. Both the global motion estimation module 604 and the global motion estimation system 672 comprise logic for encoding a current frame in accordance to the global motion compensation macroblock encoding mode. Consequently, either of these two global motion compensation systems may replace the global motion compensation module 720 comprised in the P frame mode processor 704 from Figure 7A.

However, since the global motion compensation module 720 encodes P frame macroblocks in the embodiment of Figure 7A, and since P frame macroblocks are predicted solely based on preceding frames in this embodiment, the global motion compensation module 720 only utilizes preceding frames as reference frames. Alternatively stated, global motion compensation processing performed in connection with the embodiment of Figure 7A only relies upon reference frames that precede the frame being encoded in the frame sequence. In contrast, the discussion provided above in connection with the global motion estimation module 604 and the global motion estimation system 672 contemplated that global motion compensation may be performed based on both preceding and subsequent reference frames, without limitation.

As previously discussed in connection with the global motion estimation system 672 from Figure 6F, global motion compensation

module 720 comprises three stages of logic: a prefiltering stage, an affine model estimation and motion segmentation stage, and an image warping stage. The prefiltering stage was previously described in connection with the prefiltering module 674 from Figure 6F. The prefiltering stage generally performs macroblock-based local motion filtering based on statistical attributes of motion vectors determined during local motion estimation. The affine model estimation and motion segmentation stage was described above in the context of the affine model estimation module 676 from Figure 6F. The affine model estimation process determines a set of parameters for an affine model employed in global motion estimation. The motion segmentation process filters out local motion inconsistent with the global motion estimation model. The warping stage was described above in connection with the warping module 678 from Figure 6F. The warping stage processes the current frame, a set of motion vectors derived based on the affine model, and a corresponding reference frame to determine pixel-by-pixel residual luminance and/or chrominance values for pixels in the current frame.

The data produced by the warping stage is then transmitted to the rate distortion optimizer 706, which encodes it further and compares it to data produced by other encoding modules comprised in the P frame mode processor 704 to select a preferred mode for encoding the current frame. Unlike data processed at a macroblock level by other encoding modules comprised in the P frame mode processor 704, data processed by the global motion compensation module 720 is encoded based on a frame-level affine model. To facilitate macroblock-level analysis, and to provide an adequate basis for selection of a particular encoding mode for each macroblock in the current frame, the global motion compensation module 720 partitions the encoded current frame into macroblocks prior to transmitting the corresponding data to the rate decision optimizer 706.

For each macroblock in the current frame, the rate decision optimizer 706 receives data encoded in accordance with a particular encoding mode from each module comprised in the P frame mode processor 704. For each macroblock, the rate decision optimizer 706 then determines a cost function corresponding to each encoding mode, and transmits these cost functions to a selecting module 708. The selecting module 708 compares the cost functions to select a preferred encoding mode for each macroblock in the current frame. The coding decision system 700 then utilizes the appropriate module comprised in the P frame mode processor 704 to encode each macroblock based on the selected encoding mode. For example, if comparison of cost functions by the selecting module 708 determines that a particular macroblock is preferably encoded using global motion compensation, the coding decision system 700 employs the global motion compensation module 720 to encode that particular macroblock.

(b) Bi-Directional Frame Macroblock Encoding Modes

The above description of the P frame mode processor 704 comprised in the coding decision 700 from figure 7A illustrated how a P frame may be encoded at a macroblock level using various macroblock or frame encoding modes. An analogous set of encoding modes may also be employed to encode bi-directionally predicted frames like B frames, SB frames and SSB frames. Although some of the bi-directionally predicted modes exhibit similarities with corresponding P frame encoding modes, certain differences exist. One reason for such differences is that unlike P frames, which in an embodiment of the invention are predicted exclusively in a forward direction, bi-directionally predicted frames may be predicted in both forward and backward directions, based on both preceding and subsequent frames in the frame structure.

Figure 7B illustrates a system for encoding a bi-directionally predicted frame based on selecting an encoding mode from a plurality of encoding modes according to an embodiment of the present invention. The coding decision system 730 from the embodiment of Figure 7B comprises a bi-directional frame mode processor 732, a rate distortion optimizer 706 and a selector 708. The structure and functionality of the coding decision system 730 are generally similar to the structure and functionality of the coding decision system 700 from Figure 7A. The rate distortion optimizer 706 and the selector 708 are substantially identical in both Figures 7A and 7B, but may operate on different data.

The bi-directional frame mode processor 732 comprises a number of modules that may encode a bi-directionally predicted frame based on the following encoding modes: intra, forward, backward, bi-directional (interpolation), global motion compensation (forward and backward), and direct. Each of these encoding modes provides a particular encoding format, with a corresponding cost function. The modules comprised in the bi-directionally predicted frame mode processor 732 that implement these encoding modes are further described below together with their corresponding encoding modes.

(1) Intra Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock comprised in a bi-directionally predicted frame (e.g., B, SB, or SSB frame) based on the intra encoding mode is performed substantially as previously described in connection with the intra encoding module 710 comprised in the P frame mode processor 704 illustrated in Figure 7A.

In the embodiment of Figure 7B, macroblock encoding based on the intra encoding mode is accomplished by an intra encoding module 740. The functionality and structure of the intra encoding module 740 are substantially identical to the functionality and structure of the intra encoding module 710 from Figure 7A, except that the intra encoding

module 740 operates on macroblocks comprised in bi-directionally encoded frames while the intra encoding module 710 processes P frame macroblocks. As a result of this difference, the logic comprised in the intra encoding module 710 may need to be modified to account for certain differences that may exist between the structure of the macroblock headers and/or video object plane layers associated with P frames defined in connection with conventional MPEG-4 implementations and the structure of the macroblock headers and/or video object plane layers associated with B, SB, SSB and/or other frame types provided by various aspects of the present invention.

Upon encoding a macroblock according to the intra mode, the intra encoding module 740 transmits the corresponding data to the rate distortion optimizer 706 for further processing.

(2) Forward Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock comprised in a bi-directionally predicted frame based on the forward encoding mode is performed substantially as taught in MPEG-4 in connection with encoding of B frame macroblocks. Generally, the forward encoding mode provides a framework and format for encoding a macroblock comprised in a bi-directionally predicted frame based on a reference macroblock comprised in a reference frame that precedes the bi-directionally predicted frame in the frame sequence. Consequently, the motion vectors used to define reference relationships in the forward encoding format are forward motion vectors.

In the embodiment of Figure 7B, macroblock encoding based on the forward encoding mode is accomplished by a forward encoding module 742. The functionality and structure of the forward encoding module 742 are substantially identical to the functionality and structure of the inter encoding module 712 from Figure 7A, except that the forward encoding module 742 operates on macroblocks comprised in bi-

directionally encoded frames while the inter encoding module 712 processes P frame macroblocks. Both the forward encoding mode, which applies to bi-directionally predicted frames, and the P frame encoding mode, which applies to P frames, utilize forward motion vectors. However, since certain differences may exist between the structure of the macroblock headers and/or video object plane layers associated with B frames defined in connection with conventional MPEG-4 implementations and the structure of the macroblock headers and/or video object plane layers associated with B, SB, SSB and/or other frame types provided by various aspects of the present invention, the logic comprised in the inter encoding module 712 may need to be modified accordingly.

In a particular implementation of the present invention, encoding of macroblocks comprised in SB, SSB and SSSB frames with respect to macroblocks comprised in P frames relies on a newly-introduced encoding mode denoted "Forward_ENHN." The syntax and functionality of the Forward_ENHN encoding mode is substantially identical with the syntax and functionality of the forward encoding mode.

Upon encoding a macroblock according to the forward encoding mode, the forward encoding module 742 transmits the corresponding data to the rate distortion optimizer 706 for further processing.

(3) Backward Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock comprised in a bi-directionally predicted frame based on the backward encoding mode is performed substantially as taught in MPEG-4 in connection with encoding of B frame macroblocks. Generally, the backward encoding mode provides a framework and format for encoding a macroblock comprised in a bi-directionally predicted frame based on a reference macroblock comprised in a reference frame that follows the bi-directionally predicted frame in the frame sequence. The motion vectors

used to define reference relationships in the backward encoding format are denoted "backward motion vectors."

In the embodiment of Figure 7B, macroblock encoding based on the backward encoding mode is accomplished by a backward encoding module 744. The functionality and structure of the backward encoding module 744 are substantially identical to the functionality and structure of the forward encoding module 742 from Figure 7A, except that the backward encoding module 744 employs backward motion vectors to refer to future frames in the frame sequence while the forward encoding module 742 utilizes forward motion vectors pointing to preceding frames. As a result of this difference, certain modifications may exist in the structure of the macroblock headers and/or video object plane layers employed in this embodiment in connection with the backward macroblock encoding mode.

In a particular implementation of the present invention, encoding of macroblocks comprised in SB, SSB and SSSB frames with respect to macroblocks comprised in P frames relies on a newly-introduced encoding mode denoted "Backward_ENHN." The syntax and functionality of the Backward_ENHN encoding mode is substantially identical with the syntax and functionality of the backward encoding mode.

Upon encoding a macroblock according to the backward encoding mode, the backward encoding module 744 transmits the corresponding data to the rate distortion optimizer 706 for further processing.

(4) Bi-Directional (Interpolation)

Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock comprised in a bi-directionally predicted frame based on the bi-directional (alternatively denoted "interpolation") encoding mode is performed substantially as taught in MPEG-4 in connection with

encoding of B frame macroblocks. Generally, the interpolation encoding mode provides a framework and format for encoding a macroblock comprised in a bi-directionally predicted frame based on (1) a reference macroblock comprised in a reference frame that precedes the bi-directionally predicted frame in the frame sequence, and (2) a reference macroblock comprised in a reference frame that follows the bi-directionally predicted frame in the frame sequence. The interpolation mode utilizes both forward and backward motion vectors. Since the syntax associated with encoding of B, SB, SSB and other frames provided by various aspects of the present invention may differ from the syntax conventionally employed in MPEG-4 to encode B frames, certain modifications may exist in the structure of the macroblock headers and/or video object plane layers employed in this embodiment connection with the interpolation macroblock encoding mode.

In a particular implementation of the present invention, encoding of macroblocks comprised in SB, SSB and SSSB frames based on macroblocks comprised in P frames relies on a newly-introduced encoding mode denoted "Interpolate_ENHN." The syntax and functionality of the Interpolate_ENHN encoding mode is substantially identical with the syntax and functionality of the interpolation encoding mode.

In the embodiment of Figure 7B, macroblock encoding based on the interpolation encoding mode is accomplished by an interpolation encoding module 746. Upon encoding a macroblock according to the interpolation encoding mode, the interpolation encoding module 746 transmits the corresponding data to the rate distortion optimizer 706 for further processing.

(5) Forward Global Motion Compensation Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock comprised in a bi-directionally predicted frame based on the

forward global motion compensation ("GMC-forward") encoding mode is performed substantially as previously described in connection with the global motion compensation encoding module 720 comprised in the P frame mode processor 704 illustrated in Figure 7A. Generally, the GMC-forward encoding mode provides a framework and format for encoding a macroblock comprised in a bi-directionally predicted frame based on a reference macroblock comprised in a reference frame that precedes the bi-directionally predicted frame in the frame sequence. Consequently, the motion vectors used to define reference relationships in the GMC-forward encoding format are forward motion vectors.

In the embodiment of Figure 7B, macroblock encoding based on the GMC-forward encoding mode is accomplished by a GMC-forward encoding module 748. The functionality and structure of the GMC-forward encoding module 748 are substantially identical to the functionality and structure of the global motion compensation encoding module 720 from Figure 7A, except that the GMC-forward encoding module 748 operates on macroblocks comprised in bi-directionally encoded frames while the global motion compensation encoding module 720 processes P frame macroblocks. Both the GMC-forward encoding mode, which applies to bi-directionally predicted frames, and the global motion compensation encoding mode associated with the embodiment of Figure 7A, which applies to P frames, utilize forward motion vectors.

However, since certain differences may exist between the structure of the macroblock headers and/or video object plane layers associated with P frames and the structure of the macroblock headers and/or video object plane layers associated with B, SB, SSB and/or other frame types provided by various aspects of the present invention, the logic comprised in the inter encoding module 712 may need to be modified accordingly. One implementation of the GMC-forward macroblock encoding mode is accomplished by extending the MPEG-4 Video Object Layer syntax to include a new mode denoted "BGMC." The structure and syntax of the BGMC mode is substantially similar to

the structure and syntax of the global motion estimation macroblock encoding mode taught in MPEG-4 for preceding P frames, except that the BGMC mode applies to preceding B frames in the frame structure.

Additional modifications to the MPEG-4 Video Object Plane syntax provided in an implementation of the invention include introduction of a conditional parameter CSC_BGMC_Enable into the VideoObjectLayer() header, which may be used to enable or disable GMC-forward processing. A 1-bit parameter "mcsel" specified in the Macroblock() data structure identifies the particular encoding mode employed for the respective macroblock: setting mcsel = 0 signifies that the respective macroblock may be encoded based on the forward, backward, interpolation, or direct macroblock encoding modes; and setting mcsel = 1 signifies that the respective macroblock may be encoded based on the GMC-forward or GMC-backward macroblock encoding modes.

Upon encoding a macroblock according to the GMC-forward encoding mode, the GMC-forward encoding module 748 transmits the corresponding data to the rate distortion optimizer 706 for further processing.

(6) Backward Global Motion Compensation Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock comprised in a bi-directionally predicted frame based on the backward global motion compensation ("GMC-backward") encoding mode is performed substantially as previously described in connection with the GMC-forward encoding module 748, except that the reference macroblocks that serve as an encoding basis are comprised in subsequent frames.

In the embodiment of Figure 7B, macroblock encoding based on the GMC-backward encoding mode is accomplished by a GMC-backward encoding module 750. The functionality and structure of the

GMC-backward encoding module 750 are substantially identical to the functionality and structure of the GMC-forward encoding module 748, except that the GMC-backward encoding module 750 employs backward motion vectors to refer to future frames in the frame sequence while the GMC-forward encoding module 748 utilizes forward motion vectors pointing to preceding frames. As a result of this difference, certain modifications may exist in the structure of the macroblock headers and/or video object plane layers employed in this embodiment in connection with the backward macroblock encoding mode. One such modification is introduction of a new encoding mode denoted "BGMC1." The structure and syntax of the BGMC1 mode is substantially similar to the structure and syntax of the global motion estimation macroblock encoding mode taught in MPEG-4 for subsequent P frames, except that the BGMC1 mode applies to subsequent B frames in the frame structure.

Upon encoding a macroblock according to the BGMC-backward encoding mode, the BGMC-backward encoding module 750 transmits the corresponding data to the rate distortion optimizer 706 for further processing.

(7) Direct Macroblock Encoding Mode

According to an embodiment of the invention, encoding of a macroblock comprised in a bi-directionally-predicted frame based on the direct encoding made is performed substantially as taught in MPEG-4 in connection with encoding of B frame macro blocks. Motion vectors with respect to both forward and backward reference frames are computed as a function of both an encoded motion vector "delta" and a forward motion vector corresponding to a co-located macro block in the backward reference frame.

The forward and backward reference frames may be either intra or predicted frames. In a particular case, when the current frame is a B frame, the forward and backward frames may be I or P frames.

Alternatively, when the current frame is a SB frame; the forward and backward frames may be I, P or B frames. Further, when the current frame is a SSB frame, the forward and backward frames may be I, P, B, or SB frames.

As illustrated in the embodiment of Figure 7B, a direct mode module 752 comprised in the bi-directional frame mode processor 732 receives data corresponding to a current macroblock and encodes that data in accordance with the direct encoding mode. The direct mode module 752 comprises logic for processing luminance, chrominance and/or other data associated with the current macroblock, and for encoding this data according to the direct mode format.

Upon encoding the current macroblock based on the direct macroblock encoding mode, the direct mode module 752 transmits the corresponding data to the rate distortion optimizer 706 for further processing.

iv. Rate Distortion Optimization

An aspect of the present invention provides a system for encoding a frame based on partitioning the frame into macroblocks, sequentially processing each macroblock based on various encoding modes, selecting a preferred encoding mode for each macroblock, and then encoding each macroblock in accordance with the corresponding selected preferred mode. As previously discussed, a frame mode processor may be employed to process and encode frame macroblocks based on various encoding modes. The frame mode processor may be adapted to processes different types of frame macroblocks according to preferred encoding modes. In one case, the P frame mode processor 704 from Figure 7A processes P frames macroblocks in accordance to the following encoding modes: intra, inter, inter 4V, multiple hypothesis, overlapped motion compensation, and global motion compensation. In another case, the bi-directional frame mode processor 730 from Figure 7B processes bi-directionally encoded frames (e.g., B, SB, SSB and

SSSB frames) based on the following modes: intra, forward, backward, interpolation, global motion compensation forward, global motion compensation backward, and direct.

According to an embodiment of the invention, the data corresponding to each macroblock processed by the frame mode processors described above is transmitted to a rate distortion optimizer for further processing. For each macroblock, the rate distortion optimizer determines a cost function corresponding to each encoding mode, and transmits these cost functions to a selecting module. The selecting module compares the cost functions to select a preferred encoding mode for each macroblock in the respective frame. A corresponding module comprised in the frame mode processor is then employed to encode each macroblock based on the selected preferred encoding mode.

In one case, the rate distortion optimizer that processed data produced by the frame mode processors is the rate distortion optimizer 706 from Figures 7A and 7B, and the selecting module that identifies a preferred encoding mode is the selecting module 708 from the same figures. In this case, for example, if a comparison of cost functions by the selecting module 708 determines that a particular macroblock comprised in a P frame is preferably encoded using global motion compensation, the global motion compensation module 720 comprised in the P frame mode processor 704 of Figure 7A is employed to encode that particular macroblock.

Figure 8A illustrates a system for rate distortion optimization that may be employed to select a preferred macroblock encoding mode in accordance with an aspect of the present invention. The rate distortion optimizer 800 illustrated in Figure 8A may be the rate distortion optimizer 706 from the embodiments of Figures 7A or 7B, or the rate distortion optimization subsystem 410 from the embodiment of Figure 4.

The rate distortion optimizer 800 comprises various data processing modules including an energy compactor 804, a DCT

transform 806, a quantizer 808, an entropy encoder 810, and a motion vector encoder 812. These modules cooperate to produce a measure of the data rate R corresponding to transmission of a particular macroblock encoded based on a specific encoding mode. The rate distortion optimizer 800 further comprises an inverse quantizer, an inverse DCT transform, and an inverse energy compactor that produce a measure of the distortion D corresponding to the encoded macroblock. The distortion D provides an estimate of the errors introduced by the encoding process.

An embodiment of the present invention determines a set of cost functions corresponding to the various encoding modes employed to process each macroblock in a current frame being encoded. In a particular implementation, the cost functions are determined based on a Lagrangian multiplier provided by the following formula:

$$L = D + \quad R, \qquad (10)$$

where L is the value of the Lagrangian cost function, D is the distortion, R is the data rate, and is a scaling parameter. In a particular implementation, = 1/14. For each macroblock, the rate distortion optimizer 800 determines a set of Lagrangian cost functions, each Lagrangian cost function L corresponding to a particular encoding mode. The encoding mode corresponding to the smallest Lagrangian cost function is then employed to encode the respective macroblock.

(a) Energy Compaction

Encoding and compression of video signals generally involves a transformation of data from a temporal domain to a frequency domain. A common transform function employed to convert signals to the frequency domain is the discrete cosine transform ("DCT"). In many cases, frequency transform functions of spatial data produce a poor spectral distribution of energy in the frequency domain, which may lead

to inefficient encoding of data. An aspect of the present invention provides a method for energy compaction that may improve the spectral distribution of the signal by concentrating the energy of the signal in a reduced number of frequency coefficients. An advantage of this method is that by redistributing the energy of the signal in the frequency domain, it may reduce the number of bits required to encode that signal, thereby increasing the compression of the signal.

The energy compaction method disclosed herein is preferably applied to luminance data as opposed to chrominance. This is because luminance information tends to exhibit a broader frequency spectrum. Nevertheless, this method for energy compaction may be employed to process chrominance information.

In one case, the method for energy compaction provided by this aspect of the invention is implemented within the energy compactor 804 comprised in the rate distortion optimizer 800 from Figure 8A (which corresponds to box 706 of Figures 7A and 7B). The energy compactor 804 may condition data corresponding to different macroblocks and/or different encoding modes transmitted by the frame mode processor 802 prior to frequency transformation by the DCT transform 806. An embodiment of the frame mode processor 802 is described in greater detail with regard to Figures 7A and 7B.

In one embodiment of the invention, the presence of the energy compactor 804 along the signal path between the frame mode processor 802 and the DCT transform 806 is optional. If the energy compactor 804 is not used to process a particular signal being encoded, the inverse energy compactor 818 is also bypassed.

In one embodiment, for each macroblock, the rate distortion optimizer 800 processes data corresponding to each encoding mode both in the presence and in the absence of the energy compactor 804. Consequently, for each macroblock and encoding mode, the energy compactor 804 produces two figures of merit corresponding to the data rate R: one employing energy compaction, and one without energy

compaction. Generally, bypassing of the energy compactor 804 also results in bypassing of the inverse energy compactor 818 since the two modules perform substantially-symmetrical and -reversed functions. Consequently, for each of the two data rates R, the rate distortion optimizer 800 also produces corresponding distortions D figures of merit. The rate distortion optimizer 800 then combines each of the two data rates R with the corresponding distortion D to produce two cost functions for each macroblock and encoding mode. These two cost functions are then transmitted to a decision module that evaluates the cost functions and selects the best one. In one embodiment, the selection module is the selector 708 comprised in the decision system 700 from the embodiment of Figure 7A. Consequently, according to this embodiment, a preferred encoding mode may be selected for each macroblock both in the presence and in the absence of energy compaction. Alternatively stated, encoding efficiency for each macroblock may be optimized across both encoding modes and energy compaction.

The following discussion describes the operation of two embodiments of the present invention. A first embodiment provides a method for indirect energy compaction based on a permutation of data wherein no permutation key corresponding to the permutation is explicitly transmitted to the decoder. A second embodiment provides a method for direct energy compaction with respect to an energy-normalized set of target signals, based on a permutation of the underlying data.

(1) Indirect Energy Compaction

A first embodiment provides a method for indirect energy compaction based on a permutation of the underlying data wherein no permutation key corresponding to the permutation is explicitly transmitted to the decoder. Generally, a permutation key corresponding to a reorganization of underlying data is necessary to recover the original signal. The method for energy compaction provided by this

embodiment does not require explicit transmission of the permutation key in the bitstream as overhead, thereby decreasing the global data rate. The permutation key may instead be recovered from data received at the decoder.

Figure 8B illustrates a method for indirect energy compaction based on a permutation of data, according to an aspect of the present invention. The indirect energy compaction process 830 from Figure 8B may be executed within the energy compactor 804 comprised in the rate distortion optimizer 800 from Figure 8A.

The energy compactor may process macroblocks or blocks of image data. The choice of processing a macroblock or block of variable size has to be matched to the size of the transform being used. For example, if the transform (e.g., DCT) operates on the blocks of size 8 x 8 comprising a macroblock, the energy compactor receives as input each of the respective blocks.

It is noted that the energy compactor can operate on the rows or columns of the block being processed. In this regard, columns should be construed as a form of row, orthogonal to another form of row. It is also noted that the energy compactor may also be performed in a sequence, by first processing a block row-by-row (or column-by-column), and then processing the result in a column-by-column (or row-by-row) fashion.

As illustrated in Figure 8B, the system receives data corresponding to a current block at step 832. This data may include luminance, or chrominance. The process is illustrated for the case of row operation. Normally, macroblocks processed by the rate distortion optimizer 800 are partitioned into 8x8 blocks. Consequently, the pixel row selected at step 832 is usually a block row comprising 8 pixels. Alternatively, the rate distortion optimizer 800 may operate upon 16x16 macroblocks, in which case the pixel row selected at step 832 would be a macroblock row comprising 16 pixels.

At step 834, the system also receives data from the reference macroblock or block, that corresponds to the current macroblock or block being processed. The indirect energy compaction process 830 selects a row of pixels in the reference block, corresponding to the row of pixels in the current block selected at step 832. The correspondence between rows of pixels in the current and reference blocks is normally based on a relative position within the blocks. For example, the first pixel row in the current bock corresponds to the first row in the reference block.

At step 837, the reference pixels in the reference row selected at step 834 are sorted based on a pixel attribute. In one case, the pixels are sorted monotonically based on luminance values, from the lowest value to the highest value. The permutation key associated with this reshuffling of the pixels is identified 838 and employed at step 836 to reorganize the pixels in the current row selected at step 832 in a corresponding order. It is to be noted that, in the absence of transmission errors, or through use of error resilience, and by the design of the coder/decoder system, the reference image data utilized at the encoder to encode the current macroblock or block is the same as the reference data based on which the decoder will reconstruct the current macroblock or block. Also, at the time the current macroblock or block in the current frame is processed by the decoder, the reference macroblock or block in the reference frame will have already been decoded. Thus, given the operation on the same data, the same permutation key will be obtained from the sorting of a row in the reference block at both the encoder and decoder. This process enables the extraction of the same permutation key by the encoder and decoder, without the necessity of placing the permutation for each row in a block as overhead in the bitstream. This fact may serve as a basis for subsequently decoding the current block.

At step 839, the system determines whether all rows in the current block have been processed. If additional rows remain

unprocessed, the system returns to step 832, where it selects the next row. If all rows in the current block and in the reference block have been processed and permuted, the system determines pixel-level residual values corresponding to a block prediction error. Determination of a prediction error based on subtraction of corresponding pixel attributes was previously described in connection with the global motion estimation system 672 from Figure 6F. The data produced at step 840 is then transmitted to a DCT transform 841. In one embodiment, this is the DCT transform 806 from Figure 8A.

To recover the original signal, the decoder performs a substantially-reversed process. In one case, decoding of an energy-compacted signal may be achieved using an inverse energy compactor similar to the inverse energy compactor 818. The inverse energy compactor receives a lossy block-level prediction error that comprises rows permuted as a result of sequential sorting steps performed at step 836 and steps 838 in the indirect energy compaction process 830. The inverse energy compactor may recover the permutation key for each row based on the reference block corresponding to the current block being decoded.

To recover the permutation key for each row in the current block, the inverse energy compactor analyzes the reference block that served as a basis for encoding the current block. This reference block was transmitted to the decoder and was reconstructed to help decode the current block. The decoded reference block is a representation of the reference block that was received by the indirect energy compaction process 830 at step 834. As discussed above, in the absence of transmission errors, or through use of error resilience techniques, the attributes of corresponding pixels in the reference block utilized at the encoder, and of the decoded reference block are substantially identical. Consequently, each row of the reference block may be sorted based on the same pixel attribute utilized at step 836 (e.g., luminance). The permutation obtained by sorting each row in the decoded reference

block thus provides the permutation key which will be used for the reconstruction of each corresponding row in the current block, as described below. This may be achieved without transmission of permutation keys, thereby maintaining a low data rate.

The data obtained by sorting the rows of the reference block is added pixel by pixel to the corresponding block prediction error that was extracted from the bitstream by the decoder. This process results in a row-permuted version of the current block. In order to reconstruct the current block, its rows are inverse permuted according top the corresponding row permutation keys determined from the sorting of the corresponding rows in the reference block, as described above.

The operation of the indirect energy compaction process 830 was described above in connection with permutations of rows of pixels comprised in the current and a corresponding reference block. Alternatively, the indirect energy compaction process 830 may operate on columns of pixels. In that case, at steps 832 and 834, the indirect energy compaction process 830 would select corresponding columns of pixels, and subsequent steps would operate on these columns instead of rows. In an embodiment of the invention, the rate distortion optimizer 800 configures the energy compactor 804 to sequentially process data corresponding to each block and each encoding mode based on both column and row permutations, and then the rate distortion optimizer 800 produces cost functions for each of these cases. A preferred mode of encoding each block may then be selected across all these cost functions.

(2) Direct Energy Compaction

Another embodiment of the invention provides a method for direct energy compaction with respect to an energy-normalized set of target signals, and based on a permutation of the underlying data. This method for energy compaction seeks to modify a signal prior to a frequency transformation to match certain characteristics of the

transform function, including the distribution of the frequency coefficients. The method provided by this embodiment augments the operation of the transform function to improve the spectral distribution of the frequency-converted signal by concentrating the energy of the signal in a reduced number of frequency coefficients. This method may reduce the number of bits required to encode that signal, thereby increasing the efficiency of the encoding process. It is noted that although the method is described herein in regard to predicted macroblock data, it should be recognized that the method may be used with both predicted macroblock data as well as on intra-macroblock data, both in intra frames and predicted frames.

Figure 8C illustrates a method for direct energy based on a permutation of the underlying data, according to an aspect of the present invention. The direct energy compaction process 842 from Figure 8C may be executed within the energy compactor 804 comprised in the rate distortion optimizer 800 from Figure 8A. The direct energy compaction method operates directly on the prediction error or intra macroblock or block generated by the encoder. The prediction error macroblock or block corresponds to a current macroblock or block with respect to a reference macroblock or block. The same considerations regarding matching the size of the rows or columns of the macroblocks or blocks being processed to the size of the frequency transform (e.g., DCT), apply in this case, similarly to the discussion in the previous section regarding the indirect energy compaction method. Also, the operation of the direct energy compaction process may be applied to a prediction error macroblock or block, row-by-row, column-by-column, or in a sequence of two steps, by applying a column-wise processing of the block resulting from the row processing phase.

As illustrated in Figure 8C, the system receives data corresponding to a prediction error macroblock or block at step 844. The process is illustrated for the case of a block of data. This data may include luminance and/or chrominance information. The direct energy

compaction process 842 transforms the original prediction error block with respect to set of target signals whose characteristics are described below. The target signals are organized in a target signal matrix ("TSM"). The target signal matrix may be stored at the encoder and decoder, or it may be encoded and transmitted in the bitstream.

The target signal matrix is organized to comprise a variable number M of fixed-size rows, with each row representing a target signal. Each row in the TSM has the same size as a row of the prediction error block. For example, if the prediction error block has size 8 x 8, then one row in the prediction error block will have 8 samples, and one row in the TSM (one target signal) also has 8 samples. The values comprising each of the target signals are selected such that they exhibit favorable properties with respect to the frequency transform function employed in the encoding process. In one case, the target signal matrix rows are represented by energy-normalized basis functions corresponding to the frequency transform employed. For example, if the frequency transform function is a discrete cosine transform, the rows of the target signal matrix are normalized DCT basis functions. The normalization of the target signals is done by utilizing the computed energy of each of the rows of the prediction error block, as it is described below.

At step 845, the direct energy compaction process 842 receives a target signal matrix that will serve as a basis for encoding the prediction error block. All the rows of the target signal matrix are scaled based on an estimate of the energy of the data comprised in the row of pixels selected at step 844. In a particular implementation, certain pixel attribute values (e.g., luminance values) for all pixels in the row selected at step 844 are individually squared and added together to provide a measure of the energy in the corresponding row, and all coefficients in the target signal matrix are multiplied by that value, to scale the target signals into the range of the current row being processed.

At step 846, the direct energy compaction process 842 produces all possible positional permutations for the pixels comprised in the row

selected at step 844. If the row comprises 8 pixels, the total number of permutations produced at step 844 is 40,320 (i.e., 8!).

For each permutation produced at step 846, the system processes all of the rows in the target signal matrix to determine a minimum relative cost. To achieve this, the direct energy compaction process 842 selects a permuted row at step 847 and a target row of the target signal matrix at step 848, and proceeds to step 849, where it determines a cost corresponding to the permuted row and the TSM row. In one case, the cost is determined based on a sum-of-squared differences method: corresponding pixels for the current permutation of the current row, and the TSM row are differenced, and then the differences are squared and added together to provide a cost consisting of a cumulative value, which is recorded.

At step 850, the system determines whether all the target rows in the target signal matrix have been considered as bases for comparison to the present permutation of the current row. If additional target rows remain unprocessed in the current loop, the system returns to step 848, where it selects the next target row in the target signal matrix. If the system determines at step 850 that all target rows have been considered, the direct energy compaction process 842 progresses to step 851. At step 851, the direct energy compaction process 842 has determined a set of cost functions (e.g., cumulative squared values) corresponding to the present permutation, where each cost function corresponds to a particular target row in the target signal matrix. Thus, if there are M target signals being utilized, there will be M costs corresponding to matching the current prediction error row permutation to each of the M target signals. At step 851, the system determines the minimum cost from the M costs generated for the current permutation and associates that cost with the current permutation. Essentially, this cost reflects the best match between the current permutation of the current row and the target signal matrix.

At step 852, the system determines whether all the permutations produced at step 846 have been processed for the current block row. If additional permutations remain unprocessed in the current loop, the system returns to step 847, where it selects the next permutation to be processed as described above. If the system determines at step 852 that all permutations have been considered, the direct energy compaction process 842 progresses to step 853.

At step 853, the system evaluates all costs developed at step 851 for each permutation of the current row in the block (e.g., if the size of the row is 8, then there will be 8! = 40,320 such costs). The system selects the best (minimum) cost and identifies the permutation key that produced that cost for the current block row. The system then progresses to step 854, where it applies the determined permutation key to accordingly reorganize all other rows in the current block. At step 854, all rows in the current block are permuted according to the permutation key determined at step 852.

The system then advances to step 855, where it determines a cumulative, block-level cost function corresponding to the current block row being processed. To obtain this cumulative, block-level cost function, the system may proceed as follows. The permutation determined at step 853 for the current row, is applied to all the other rows in the block prediction error. The system then operates on the rows of the resulting block data to compute the best match between each of its rows and the rows from the TSM. The best match to TSM for each block row can be determined based on a cost function similar to the one discussed above (e.g., least squares). The system then adds these best row-level cost functions corresponding to the all the rows in the block to obtain a cumulative, block-level cost function corresponding to the current block row. This permutation of the current row in the prediction error block and its corresponding block-level cumulative cost determined as described above are memorized.

The system then progresses to step 856, where it assesses whether all the rows in the current block have been processed. If additional rows remain unprocessed, the system returns to step 844, where it selects the next block row to be processed as described above. If the system determines at step 856 that all block rows have been processed, however, the direct energy compaction process 842 progresses to step 857. At step 857, the system has identified a set of permutation keys corresponding to each row in the prediction block that produce the best cumulative, block-level costs determined at step 855.

At step 857, the direct energy compaction process 842 compares all cumulative, block-level costs determined as described above at step 855, and identifies the permutation key corresponding to the best (minimum) cost. The system then employs this particular permutation key to reorganize each row in the prediction error block, at step 858. This permuted version of the prediction error block may be better suited for the subsequent frequency transformation, and may provide a more compact energy distribution in the frequency domain. To facilitate reconstruction of the original block, the permutation key utilized to encode the prediction error block is transmitted to the decoder. The decoder applies this permutation key to each of the rows or columns (according to the encoder's selection) of the lossy prediction error block it extracts from the bitstream, which it then adds to the reference block from the already decoded reference frame in order to reconstruct the current block.

v. Entropy Encoding

As illustrated in Figure 8A and discussed above in connection with the energy compactor 804, data transmitted by the frame mode processor 802 is optionally processed by the energy compactor 804, and is then received by the DCT transform 806. The DCT transform 806 processes this data at a macroblock, block, or sub-block level to convert it to a frequency domain. The DCT transform 806 may be a

conventional discrete cosine transformer commonly employed in image signal processing. In one embodiment, the DCT transform 806 processes prediction error data comprising 8x8 blocks of residual luminance values and converts this data into 8x8 blocks of corresponding frequency coefficients.

The 8x8 blocks of corresponding frequency coefficients produced by the DCT transform 806 are then transmitted to a quantizer 808. The quantizer 808 quantizes the frequency coefficients and converts them into corresponding quantized values. The quantizer 808 may be a conventional quantizer commonly employed in video compression. In a particular implementation, the quantizer 808 is an H263 or MPEG-4 quantizer. In one embodiment, for 8x8 blocks of frequency coefficients, the quantizer 808 produces corresponding sets of 64 quantized values, wherein each quantized value is an integer between 0 and 4,095.

The sets of quantized values are then transmitted to an entropy encoder 810. In addition to each such set of quantized values, the entropy encoder 810 also receives a corresponding set of motion vectors transmitted by the frame mode processor 802 and encoded by a motion vector encoder 812. The motion vector encoder 812 may be a conventional MPEG-4 motion vector encoder.

The entropy encoder 810 converts data received from the quantizer 808 and the motion vector encoder 812 into a bitstream suitable for transmission via a communication channel or for storage on a storage medium. In one embodiment of the invention, the entropy encoder 810 is a conventional MPEG-4 run-length encoder using variable length codes.

In one case, the entropy encoder 810 processes the incoming data to derive a transmission rate R for the corresponding bitstream without fully converting the data into a bitstream. To achieve this, the entropy encoder 810 partially encodes the data into a corresponding bitstream to estimate the number of bits that would be necessary to transmit the data if the data were fully encoded based on the encoding

mode under consideration and under the particular applicable conditions. The data rate R may then provide a basis for performing a coding decision as previously discussed in connection with the rate distortion optimizer 800 illustrated in Figure 8A and equation 10.

An aspect of the invention provides a method for encoding data comprised in a coefficient matrix into a bitstream by processing coefficients in a zig-zag order based on an adaptive context. The context is adaptively determined based on the direction of the zig-zag scan such that for each coefficient under consideration, the context comprises the preceding coefficient but excludes the subsequent coefficient. An advantage of the present aspect of the invention is that this method employs information regarding neighboring coefficients to encode a particular coefficient, thereby increasing the coding efficiency and accuracy.

Figure 8D illustrates an adaptive context that may be serve as a basis for encoding data into a bitstream in a zig-zag scanning pattern, according to an aspect of the present invention. In one embodiment, the entropy encoder 810 from Figure 8A relies on the adaptive context illustrated in Figure 8D to encode data transmitted by the quantizer 808 into a bitstream.

As shown in Figure 8D, an 8x8 pixel block comprises 64 pixels disposed in a two-dimensional rectangular matrix. As illustrated in Figure 860, the pixels are scanned in a zig-zagged order and are encoded sequentially based on an adaptive context. The adaptive context comprises neighboring pixels that have already been encoded and provides a basis for encoding a current pixel. The pixel that is scheduled to be encoded immediately after the current pixel based on the zig-zagged scanning pattern is explicitly excluded from the context.

Figure 8D shows a set of 10 numbered coefficients in the macroblock 860. The coefficients are numbered according to the zig-zag pattern in which they are processed during encoding. Coefficient 5 is currently being encoded. The context 862 for coefficient 5 comprises

neighboring coefficients that have already been processed: coefficients 1, 2, 3 and 4. Coefficient 6 is scheduled to be encoded immediately after the current coefficient 5 and therefore is explicitly excluded from the context 862. Additional neighboring coefficients that have not been processed yet (i.e., coefficients 8 and 9) are also excluded from the context 862. Applying these rules to a further example, the context for encoding pixel 8 would consists of coefficients 2, 5, 6 and 7.

The context for a coefficient disposed proximally to one or more edges of the block is modified consistent with the underlying principle that a context may only include coefficients that have already been encoded. For example, coefficient 1 has no actual context, but is conventionally defined to include coefficient 1. Similarly, the context for coefficient 2 consists solely of coefficient 1. Analogously, the context for coefficient 3 consists of coefficient 1 and 2. Contexts for other coefficients in the block 860 may be similarly determined.

In a particular implementation, the context of a current coefficient may be expanded to comprise additional coefficients, including coefficients that have already been processed but that are not immediate neighbors of the current coefficient. For example, the context of coefficient 9 from Figure 8D may be expanded to include coefficients 1-8, but not coefficient 10. Increasing the number of coefficients comprised in a particular context may help improve the accuracy and efficiency of the encoding process.

Once the entropy encoder 810 determines a context for a particular coefficient, the entropy encoder 810 evaluates a corresponding context sum by adding together the context values of the respective context coefficients. To determine context values for individual coefficients in the current block, the entropy encoder 810 processes the quantized coefficients produced and transmitted by the quantizer 808 according to the following set of rules: if a quantized coefficient is 0 or 1, the context value for that coefficient is set equal to its value; otherwise, the context value is set to 2. Alternatively stated,

coefficient context values consist of corresponding quantized coefficients, but are capped to a maximum value of 2. Although quantized coefficients are integers that may vary between 0 and 4,095, typically the values of quantized coefficients tend to be relatively small.

Once the entropy encoder 810 has determined context values for the coefficients, a context sum C_{SUM} is determined for the current coefficient based on the following formula:

$$C_{SUM} = a_1 C_1 + a_2 C_2 + a_3 C_3 + a_4 C_4, \qquad (11)$$

where a_1, a_2, a_3 and a_4 are a set of weights and C_1, C_2, C_3 and C_4 are context values. For coefficients located proximally to edges of the block, one or more of the context values may be zero.

The values of the weights a_1, a_2, a_3 and a_4 are selected such that each possible combination of context values provides a unique context sum. If each of the four context values C_1, C_2, C_3 and C_4 may only take three values (i.e., 0, 1 or 2), a maximum of 81 unique combinations of context coefficients may occur (i.e., 3^4 combinations). In a particular implementation, $a_1 = 1$, $a_2 = 3$, $a_3 = 9$ and $a_4 = 27$. In this implementation, the context sum C_{SUM} may take any value between 0 (for $a_1 = a_2 = a_3 = a_4 = 0$) and 80 (for $a_1 = a_2 = a_3 = a_4 = 2$), for a total of 81 possible unique values. Consequently, in this implementation, each possible combination of context coefficients may be identified by a unique context sum.

In one embodiment, for each coefficient in the current block, the entropy encoder 810 employs a specific probability density function to perform arithmetic coding on the quantized value corresponding to that coefficient. Each unique context sum is associated with a corresponding probability density function. Once the entropy encoder 810 determines a specific context sum C_{SUM} for a particular coefficient, the entropy encoder 810 retrieves the probability density function corresponding to that context sum value. In one embodiment, the entropy encoder 810

performs a look up into a table to retrieve probability density functions corresponding to specific context sum values.

Once the entropy encoder 810 retrieves a particular probability density function corresponding to a specific coefficient context combination, the entropy encoder 810 utilizes that probability density function as a basis for arithmetically encoding the quantized value associated with that coefficient to produce a corresponding encoded bitstream. The entropy encoder 810 utilizes the process disclosed herein to sequentially process all coefficients in the current block or macroblock, producing a set of corresponding quantized value bitstreams. The entropy encoder 810 then multiplexes this set of quantized value bitstreams with a motion vector bitstream corresponding to one or more motion vectors associated with the current block provided by the motion vector encoder 812 to produce a final bitstream encoding the current macroblock. This final bitstream is a compressed representation of the corresponding original block and is suitable for efficient transmission via a communication channel and/or efficient storage on a storage medium.

D. Variables

The following identifies a number of variables that are employed in one implementation of the invention to facilitate data processing in accordance with various aspects and embodiments of the present invention. Some or all of these variables may be utilized in connection with, and as an extension to, conventional MPEG-4 syntax.

Variable	Values
(PVOP Macroblocks)	
P MacroBlock Type	5 – OBMC
	6 – OBMC (Q modulated)
	7 – Multiple Hypothesis
	8 – Multiple Hypothesis (Q modulated)
Reference Frame ID	0-7
MH Reference Frame ID	0-7
MH Vector Data	Same as MPEG-4 MB vector data
(BVOP Macroblocks)	
B MacroBlock Type	4 – INTRA
	5 – GMC Backward
	6 – GMC Forward

1. C. Post-processing Methods

A variety of methods may be employed to process decoded images. These methods are most commonly used to improve the resulting visual effect of the decoded images. For example, these methods may be employed to reduce contours in decoded images and to reduce the visual appearance of artifacts.

Contours can result from high quantization in the encoding process. They usually appear in very smooth regions in decoded images. Small quantization steps can be used to remove contours but will increase the coding bit rate. In order to alleviate or reduce of contouring effects without increasing the coding bit rate, it is more desirable to perform post-processing to reduce the contours by dithering. Dithering is a process in which random noise is added to decoded images. Dithering methods may apply spatially uniform random noise to images. The main drawback of existing dithering methods is that the

added noise can be strongly felt. It is thus desirable for dithering to be performed in a way which reduces the visual effect of noise.

In one embodiment according to the present invention, a post processing dithering method is provided for reducing contours in a decoded image. According to the embodiment, the method comprises: taking a decoded image; for blocks of the decoded image, computing local variances for each block; using the computed local variances to determine smooth regions of the decoded image; segmenting the detected smooth regions; randomly selecting pixels within each smooth region; and adding random noise to the selected pixels. Reducing contours in decoded images by dithering is preferably conducted after deblocking and/or deringing. Desirably, the above embodiment is able to achieve the desired reduction of contours without creating undesirable visual effects.

The present invention also provides a directional filter for removing blocking artifacts in decoded images and a regularization filter which provides further smoothing.

In regard to the directional filter, in one embodiment, a method is provided for reducing block artifacts in decoded images comprising: taking a decoded block having edge pixels defining a perimeter of the block; and for each edge pixel: taking an edge pixel as a current pixel, determining a local variance for the current pixel by comparing the current pixel to pixels neighboring the current pixel, determining absolute differences between two neighboring pixels of the current pixel a) normal to the edge, b) in a positive diagonal direction relative to the edge, and c) in a negative diagonal direction relative to the edge, constructing a pixel-based directional filter based on the determined local variance and the determined absolute differences, and calculating a new value for the current pixel using the constructed pixel-based directional filter.

In regard to the regularization filter, in another embodiment, a method is provided for smoothing artifacts within a decoded image comprising: constructing a cost function including a closeness term and

a smoothness term; determining a system of linear equations by minimizing the cost function; and iteratively solving the equations for smoothed values for all pixels.

These two post-processing methods serve to reduce blocking artifacts in decoded images. The method employing a directional filter is generally performed first on pixels at block edges to remove obvious blocking artifacts. The regularization filter is then typically performed on all pixels to smooth the entire image.

The decoded images are usually degraded in the process of encoding. The most typical artifacts are blocking effect, ringing and contouring. Blocking artifact is introduced as the result of separate 8x8 block encoding and quantization and is the main annoying artifact.

Previous deblocking methods have not been sufficiently efficient to be implemented in real time. By contrast, the methods of the present invention provide a way to remove the blocking artifacts efficiently and also make it possible for real-time application.

In order to better appreciate the directional filter method, consider a case in the following diagram. The nine pixels 1, 2, 3, 4, 5, 6, 7, 8, and 9 are located in

two separate 8x8 blocks where 1, 2 and 3 are the pixels at the upper block edge and 4, 5 and 6 are the pixels at the lower block edge. The nine pixels have the values

$a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9$

A directional filter is constructed and applied on the pixel 5.

1	2	3
4	**5**	6
7	8	9

The local variance for pixel 5 is firstly calculated based on these nine pixels. Then the directional filter can be expressed by the following equation

$$a_5^{'} = 0.5w\left(\frac{a_1 + a_9}{1 + kw|a_1 - a_9|} + \frac{a_2 + a_8}{1 + kw|a_2 - a_8|} + \frac{a_3 + a_7}{1 + kw|a_3 - a_7|}\right)$$

$$+ a_5\left(1 - \frac{w}{1 + kw|a_1 - a_9|} - \frac{w}{1 + kw|a_2 - a_8|} - \frac{w}{1 + kw|a_3 - a_7|}\right)$$

where

$$k = c \times \mathrm{var}\,iance$$

c is a constant and w =0.25. The new value for the pixel 5 is calculated from this equation. The directional filter is applied on all pixels at block edges in both horizontal and vertical directions.

In order to better appreciate the regularization filter method, assume

$f_{i,j}^{\ 0}$ is the original value for the pixel (i, j) and

$f_{i,j}$ is the filtered value for the pixel (i, j). Then a cost function can be constructed as follows

$$C(f_{i,j}) = \sum_i \sum_j \left(f_{i,j} - f_{i,j}^{\ 0}\right)^2 + \lambda \sum_i \sum_j \left[\left(f_{i,j} - f_{i-1,j}\right)^2 + \left(f_{i,j} - f_{i,j-1}\right)^2\right]$$

where λ is a parameter that can be used to adjust the smoothing degree. Larger value is for more smoothing and smaller value is for less smoothing. The first term in the right side of the equation is the

closeness constraint that represents the fidelity of the filtered to the original data. The second term in the right side imposes a smoothness constraint. The minimization of the cost function can be found by setting the first derivative to zero with respect to each unknown

$$\frac{\partial C(f_{i,j})}{\partial f_{i,j}} = (1+4\lambda) f_{i,j} - \lambda(f_{i-1,j} + f_{i+1,j} + f_{i,j-1} + f_{i,j+1}) - f_{i,j}^{\ 0} = 0$$

which results in a system of simultaneous equations

$$Af = b$$

The matrix A is symmetric, positive, definite, sparse and banded. Various algorithms can be used to solve the equations.

It is noted that in large resolution and high video quality applications, the directional filter is typically only used in the smooth region of decoded images. In low bit rate applications, both directional filter and regularization filter may be used for the entire decoded images.

The foregoing description of a preferred embodiment of the invention has been presented for purposes of illustration and description. It is not intended to be exhaustive or to limit the invention to the precise forms disclosed. Many modifications and variations will be apparent to practitioners skilled in this art. It is intended that the scope of the invention be defined by the following claims and their equivalents.

We claim:

1. Computer readable medium comprising:

data encoding a sequence of frames of video in a compressed format, the encoded video frames comprising intra frames which do not rely on another frame to encode an image for that frame, predicted frames which rely on a preceding intra frame or a preceding predicted frame to encode an image for that frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein a periodicity of intra frames and/or predicted frames varies within the encoded sequence of video frames; and

logic for decompressing the data into a less compressed video format.

2. Computer readable medium according to claim 1 wherein the periodicity of intra frames varies within the encoded sequence of video frames.

3. Computer readable medium according to claim 1 wherein the periodicity of predicted frames varies within the encoded sequence of video frames.

4. Computer readable medium according to claim 1 wherein the periodicity of bi-directionally predicted frames varies within the encoded sequence of video frames.

5. Computer readable medium according to claim 1 wherein one, two, three, four, five, six, seven, eight, or nine bi-directionally predicted frames are positioned between successive intra or predicted frames in the encoded video sequence.

6. Computer readable medium comprising:

data encoding a sequence of frames of video in a compressed format, the encoded video frames comprising intra frames which do not rely on another frame to encode an image for that frame, predicted frames which rely on a preceding intra frame or a preceding predicted frame to encode an image for that frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein a periodicity of intra frames and/or predicted frames varies within the encoded sequence of video frames based on a relative efficiency of encoding a given frame as different frame types; and

logic for decompressing the data into a less compressed video format.

7. Computer readable medium according to claim 6 wherein the periodicity of intra frames and/or predicted frames varies within the encoded sequence of video frames based on a combination of the relative efficiency of encoding a given frame as different frame types and an image quality cost for encoding the given frame as the different frame types.

8. Computer readable medium according to claim 6 wherein the periodicity of intra frames varies within the encoded sequence of video frames.

9. Computer readable medium according to claim 6 wherein the periodicity of predicted frames varies within the encoded sequence of video frames.

10. Computer readable medium according to claim 6 wherein the periodicity of bi-directionally predicted frames varies within the encoded sequence of video frames.

11. Computer readable medium according to claim 6 wherein one, two, three, four, five, six, seven, eight, or nine bi-directionally predicted frames are positioned between successive intra or predicted frames in the encoded video sequence.

12. Computer readable medium comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding intra frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as an intra frame or another frame type.

13. Computer readable medium according to claim 12 wherein the periodicity of intra frames within the encoded sequence of video frames encoded by the logic varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as an intra frame or another frame type.

14. Computer readable medium according to claim 12 wherein the logic analyzes whether it is more efficient to encode some of the frames as an intra frame or a predicted frame.

15. Computer readable medium according to claim 12 wherein the logic analyzes whether it is more efficient to encode some of the frames as an intra frame or a bi-directionally predicted frame.

16. Computer readable medium according to claim 12 wherein the logic analyzes whether it is more efficient to encode some of the frames as a predicted frame or a bi-directionally predicted frame.

17. Computer readable medium according to claim 12 where the logic encodes one, two, three, four, five, six, seven, eight, or nine bi-directionally predicted frames between adjacent intra or predicted frames in the encoded video sequence.

18. Computer readable medium according to claim 12 wherein the logic uses more than one preceding frame to evaluate the efficiency of encoding a given frame as an intra frame.

19. Computer readable medium according to claim 12 wherein the logic uses more than one preceding frame selected from the group of intra frames and predicted frames to evaluate the efficiency of encoding a given frame as an intra frame.

20. Computer readable medium according to claim 12 wherein the logic employs a whole frame or a still image encoder to encode intra frames.

21. Computer readable medium according to claim 12 wherein the logic employs a JPEG 2000 encoder to encode intra frames.

22. Computer readable medium according to claim 12 wherein the logic employs a wavelet-based encoder to encode intra frames.

23. Computer readable medium according to claim 12 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by fully encoding the frame as both an intra frame and another frame type.

24. Computer readable medium according to claim 12 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by fully encoding the frame as both an intra frame and a predicted frame type.

25. Computer readable medium according to claim 12 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by fully encoding the frame as both an intra frame and a bi-directional predicted frame type.

26. Computer readable medium according to claim 12 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by fully encoding the frame as an intra frame, a predicted frame and a bi-directional predicted frame type.

27. Computer readable medium according to claim 12 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by only partially encoding the frame as both an intra frame and another frame type.

28. Computer readable medium according to claim 12 wherein the logic evaluates whether it is more efficient to encode a given frame

as an intra frame by only partially encoding the frame as both an intra frame and a predicted frame type.

29. Computer readable medium according to claim 12 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by only partially encoding the frame as both an intra frame and a bi-directional predicted frame type.

30. Computer readable medium according to claim 12 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by only partially encoding the frame as an intra frame, a predicted frame and a bi-directional predicted frame type.

31. Computer readable medium according to claim 12 wherein the logic for encoding video in a compressed format is adapted to encode video frames pre-encoded in an MPEG format.

32. Computer readable medium comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding predicted frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as a predicted frame or another frame type.

33. Computer readable medium according to claim 32 wherein the periodicity of predicted frames within the encoded sequence of video frames encoded by the logic varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as a predicted frame or another frame type.

34. Computer readable medium comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding bi-directional predicted frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as a bi-directional frame or another frame type.

35. Computer readable medium according to claim 34 wherein the periodicity of bi-directional predicted frames within the encoded sequence of video frames encoded by the logic varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as a bi-directional predicted frame or another frame type.

36. An encoder for encoding video in a compressed format, the encoder comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more

compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding intra frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as an intra frame or another frame type.

37. An encoder according to claim 36 wherein the periodicity of intra frames within the encoded sequence of video frames encoded by the logic varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as an intra frame or another frame type.

38. An encoder according to claim 36 wherein the logic analyzes whether it is more efficient to encode some of the frames as an intra frame or a predicted frame.

39. An encoder according to claim 36 wherein the logic analyzes whether it is more efficient to encode some of the frames as an intra frame or a bi-directionally predicted frame.

40. An encoder according to claim 36 wherein the logic analyzes whether it is more efficient to encode some of the frames as a predicted frame or a bi-directionally predicted frame.

41. An encoder according to claim 36 where the logic encodes one, two, three, four, five, six, seven, eight, or nine bi-directionally

predicted frames between adjacent intra or predicted frames in the encoded video sequence.

42. An encoder according to claim 36 wherein the logic uses more than one preceding frame to evaluate the efficiency of encoding a given frame as an intra frame.

43. An encoder according to claim 36 wherein the logic uses more than one preceding frame selected from the group of intra frames and predicted frames to evaluate the efficiency of encoding a given frame as an intra frame.

44. An encoder according to claim 36 wherein the logic employs a whole frame or a still image encoder to encode intra frames.

45. An encoder according to claim 36 wherein the logic employs a JPEG 2000 encoder to encode intra frames.

46. An encoder according to claim 36 wherein the logic employs a wavelet-based encoder to encode intra frames.

47. An encoder according to claim 36 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by fully encoding the frame as both an intra frame and another frame type.

48. An encoder according to claim 36 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by fully encoding the frame as both an intra frame and a predicted frame type.

49. An encoder according to claim 36 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by fully encoding the frame as both an intra frame and a bi-directional predicted frame type.

50. An encoder according to claim 36 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by fully encoding the frame as an intra frame, a predicted frame and a bi-directional predicted frame type.

51. An encoder according to claim 36 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by only partially encoding the frame as both an intra frame and another frame type.

52. An encoder according to claim 36 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by only partially encoding the frame as both an intra frame and a predicted frame type.

53. An encoder according to claim 36 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by only partially encoding the frame as both an intra frame and a bi-directional predicted frame type.

54. An encoder according to claim 36 wherein the logic evaluates whether it is more efficient to encode a given frame as an intra frame by only partially encoding the frame as an intra frame, a predicted frame and a bi-directional predicted frame type.

55. An encoder according to claim 36 wherein the logic for encoding video in a compressed format is adapted to encode video frames pre-encoded in an MPEG format.

56. An encoder for encoding video in a compressed format, the encoder comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding predicted frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as a predicted frame or another frame type.

57. An encoder according to claim 56 wherein the periodicity of predicted frames within the encoded sequence of video frames encoded by the logic varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as a predicted frame or another frame type.

58. An encoder for encoding video in a compressed format, the encoder comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-

directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the logic encoding bi-directional predicted frames in the encoded sequence of video frames with variable periodicity within the encoded video sequence based at least in part on an analysis of whether it is more efficient to encode at least some frames as a bi-directional frame or another frame type.

59. An encoder according to claim 57 wherein the periodicity of bi-directional predicted frames within the encoded sequence of video frames encoded by the logic varies based at least in part on a combination of the coding efficiency analysis and an image quality cost analysis for encoding at least some frames as a bi-directional predicted frame or another frame type.

60. Computer readable medium comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein the logic comprises decision logic which evaluates whether to encode a given frame as an intra frame or another frame type.

61. Computer readable medium according to claim 60 wherein the decision logic evaluates whether to encode a given frame as an intra frame or another frame type based at least in part on whether it is more coding efficient to encode the given frame as an intra frame or another frame type.

62. Computer readable medium according to claim 60 wherein the decision logic evaluates whether to encode a given frame as an intra frame or another frame type based at least in part on a combination of whether it is more coding efficient to encode the given frame as an intra frame or another frame type coding efficiency and an image quality cost function.

63. Computer readable medium according to claim 60 wherein the decision logic comprises logic for evaluating whether to encode a given frame as an intra frame or a predicted frame.

64. Computer readable medium according to claim 60 wherein the decision logic comprises logic for evaluating whether to encode a given frame as an intra frame or a bi-directional predicted frame.

65. Computer readable medium according to claim 60 wherein the decision logic comprises logic for evaluating whether to encode a given frame as an intra frame, a predicted frame, or a bi-directional predicted frame.

66. Computer readable medium according to claim 60 wherein the decision logic uses more than one preceding frame to evaluate whether to encode a given frame as an intra frame or another frame type.

67. Computer readable medium according to claim 60 wherein the decision logic uses more than one preceding frame selected from the group of intra frames and predicted frames to evaluate whether to encode a given frame as an intra frame or another frame type.

68. Computer readable medium according to claim 60 wherein the decision logic evaluates whether to encode a given frame as an intra frame or another frame type by fully encoding the given frame as both an intra frame and another frame type.

69. Computer readable medium according to claim 60 wherein the decision logic evaluates whether to encode a given frame as an intra frame or another frame type by only partially encoding the given frame as both an intra frame and another frame type.

70. Computer readable medium according to claim 60 wherein the logic for encoding video in a compressed format is adapted to encode video frames pre-encoded in an MPEG format.

71. An encoder for encoding video in a compressed format, the encoder comprising:

logic for receiving a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein the logic comprises decision logic which evaluates whether to encode a given frame as an intra frame or another frame type.

72. An encoder according to claim 71 wherein the decision logic evaluates whether to encode a given frame as an intra frame or another frame type based at least in part on whether it is more coding efficient to encode the given frame as an intra frame or another frame type.

73. An encoder according to claim 71 wherein the decision logic evaluates whether to encode a given frame as an intra frame or another frame type based at least in part on a combination of whether it is more coding efficient to encode the given frame as an intra frame or another frame type coding efficiency and an image quality cost function.

74. An encoder according to claim 71 wherein the decision logic comprises logic for evaluating whether to encode a given frame as an intra frame or a predicted frame.

75. An encoder according to claim 71 wherein the decision logic comprises logic for evaluating whether to encode a given frame as an intra frame or a bi-directional predicted frame.

76. An encoder according to claim 71 wherein the decision logic comprises logic for evaluating whether to encode a given frame as an intra frame, a predicted frame, or a bi-directional predicted frame.

77. An encoder according to claim 71 wherein the decision logic uses more than one preceding frame to evaluate whether to encode a given frame as an intra frame or another frame type.

78. An encoder according to claim 71 wherein the decision logic uses more than one preceding frame selected from the group of intra frames and predicted frames to evaluate whether to encode a given frame as an intra frame or another frame type.

79. An encoder according to claim 71 wherein the decision logic evaluates whether to encode a given frame as an intra frame or another frame type by fully encoding the given frame as both an intra frame and another frame type.

80. An encoder according to claim 71 wherein the decision logic evaluates whether to encode a given frame as an intra frame or another frame type by only partially encoding the given frame as both an intra frame and another frame type.

81. An encoder according to claim 71 wherein the logic for encoding video in a compressed format is adapted to encode video frames pre-encoded in an MPEG format.

82. A computer executed method for encoding a sequence of frames of video in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or a preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, the computer executed method comprising:

analyzing whether encoding a given frame of the sequence as an intra frame or another frame type is more coding efficient;

using results from the analysis to decide whether to encode the given frame as an intra frame or another frame type; and

encoding the given frame as an intra frame or another frame type based on the decision.

83. A computer executed method according to claim 82 wherein the another frame type is selected from the group consisting of predicted frame and bi-directional predicted frame.

84. A computer executed method according to claim 82 wherein the another frame type is a predicted frame.

85. A computer executed method according to claim 82 wherein the another frame type is a bi-directional predicted frame.

86. A computer executed method according to claim 82, wherein

the method further comprises analyzing an image quality cost associated with encoding the given frame as an intra frame or another frame type; and

using results from the analysis to decide whether to encode the given frame as an intra frame or another frame type comprises both results from the analysis of whether it is more coding efficient to encode the given frame as an intra frame or another frame type and the image quality cost results.

87. A computer executed method according to claim 82, wherein the analysis regarding whether encoding a given frame as an intra frame or another frame type is more coding efficient is performed using more than one preceding frame selected from the group of intra frames and predicted frames as potential reference frames for the given frame.

88. A computer executed method according to claim 82, wherein the analysis regarding whether encoding a given frame as an intra frame or another frame type is performed by only partially encoding the given frame as both an intra frame and at least one other frame type.

89. A computer executed method according to claim 82, wherein the analysis regarding whether encoding a given frame as an intra frame or another frame type is performed by fully encoding the given frame as both an intra frame and at least one other frame type.

90. A computer executed method according to claim 82, wherein the sequence of frames of video to be encoded are already in an MPEG format.

91. A method for encoding a sequence of frames of video comprising:

encoding a first group of frames as intra frames which do not rely on another frame to encode an image for that frame;

encoding a second group of frames as predicted frames which rely on a preceding intra frame or a preceding predicted frame to encode an image for that frame;

encoding a third group of frames as bi-directional predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame; and

encoding a fourth group of frames as super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame, and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, wherein at least a portion of the super bi-directional predicted frames are encoded with reference to at least one bi-directional frame.

93. A method according to claim 92 wherein at least a portion of the super bi-directional predicted frames rely on a preceding bi-directional frame.

94. A method according to claim 92 wherein at least a portion of the super bi-directional predicted frames rely on a subsequent bi-directional frame.

95. A method according to claim 92 wherein at least a portion of the super bi-directional predicted frames rely upon a preceding intra

frame, predicted frame, or bi-directional frame which is not a frame immediately preceding the super bi-directional predicted frame.

96. A method according to claim 92 wherein at least a portion of the super bi-directional predicted frames rely upon a subsequent intra frame, predicted frame, or bi-directional frame which is not a frame immediately following the super bi-directional predicted frame.

97. Computer readable medium comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame.

98. Computer readable medium according to claim 97, further comprising logic for determining an available bandwidth for transmitting video in a compressed format and logic for transmitting video with or without super bi-directional predicted frames based on the determined available bandwidth.

99. Computer readable medium comprising logic for decoding video from a compressed format, the logic comprising:

logic for decoding intra frames that do not rely on another frame to encode an image for that frame;

logic for decoding predicted frames that rely on a preceding intra frame or predicted frame,

logic for decoding bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame; and

logic for decoding super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame.

100. A method for decoding video from a compressed format, the method comprising:

decoding a group of intra frames which do not rely on another frame to encode an image for that frame;

decoding a group of predicted frames that rely on a preceding intra frame or predicted frame;

decoding a group of bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame; and

decoding a group of super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame, and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame.

101. Computer readable medium comprising:

logic for encoding video in a compressed format, the logic taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or preceding predicted frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to

encode an image for that frame, the logic encoding intra frames using a JPEG 2000 encoder.

102. Computer readable medium encoding logic for detecting scene changes in a sequence of frames of video, the logic comprising:

logic for encoding video in a compressed format by taking a sequence of frames of video and encoding the frames in a more compressed format that comprises intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or preceding frame, and bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, wherein the logic evaluates whether to encode a given frame as an intra frame or a predicted frame based at least in part on a relative coding efficiency between encoding the frame as an intra frame or a predicted frame; and

logic which identifies a frame as being a beginning of a scene change based, at least in part, on the frame being encoded as an intraframe.

103. Computer readable medium according to claim 102 wherein the decision logic evaluates whether to encode a given frame as an intra frame or a predicted frame based at least in part on a combination of whether it is more coding efficient to encode the given frame as an intra frame or a predicted frame and an image quality cost function.

104. Computer readable medium according to claim 102 wherein the decision logic evaluates whether to encode a given frame as an intra frame or a predicted frame by fully encoding the given frame as both an intra frame and a predicted frame.

105. Computer readable medium according to claim 102 wherein the decision logic evaluates whether to encode a given frame as an intra frame or a predicted frame by only partially encoding the given frame as both an intra frame and a predicted frame.

106. A method for transmitting video in a compressed format, the compressed video comprising a sequence of frames that comprise intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, the method comprising:

determining an amount of bandwidth required to transmit the sequence of frames with or without use of super bi-directionally predicted frames;

determining an amount of bandwidth available at a given time to transmit the sequence of frames with or without use of super bi-directionally predicted frames; and

transmitting the video data with or without use of super bi-directionally predicted frames based at least in part on whether the bandwidth available at the given time is sufficient to transmit the sequence of frames without the use of super bi-directionally predicted frames.

107. A method according to claim 106 wherein the super bi-directionally predicted frames are used is also based in part on an image quality cost function associated with using the super bi-directionally predicted frames.

108. A method according to claim 106 wherein determining the amount of bandwidth required and determining the amount of bandwidth available is performed continuously as the video data is transmitted, the video data being transmitted both with and without super bi-directionally predicted frames over time based on whether the bandwidth available at the given time is sufficient to transmit the sequence of frames without the use of super bi-directionally predicted frames.

109. A method according to claim 106 wherein determining the amount of bandwidth required and determining the amount of bandwidth available is performed periodically as the video data is transmitted, the video data being transmitted both with and without super bi-directionally predicted frames over time based on whether the bandwidth available at the given time is sufficient to transmit the sequence of frames without the use of super bi-directionally predicted frames.

110. Computer readable medium for use in a method for transmitting video in a compressed format, the compressed video comprising a sequence of frames that comprise intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, the method comprising:

logic for determining an amount of bandwidth required to transmit the sequence of frames with or without use of super bi-directionally predicted frames;

logic for determining an amount of bandwidth available at a given time to transmit the sequence of frames with or without use of super bi-directionally predicted frames; and

logic for causing the video data to be transmitted with or without use of super bi-directionally predicted frames based at least in part on whether the bandwidth available at the given time is sufficient to transmit the sequence of frames without the use of super bi-directionally predicted frames.

111. A method for decoding video transmitted in a compressed format, the compressed video comprising a sequence of frames that comprise intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, the method comprising:

determining whether super bi-directionally predicted frames are being transmitted; and

if super bi-directionally predicted frames are being transmitted, decoding the super bi-directionally predicted frames, or

if super bi-directionally predicted frames are not being transmitted, decoding transmitted intra frames, predicted frames, and bi-directionally predicted frames, and producing additional frames based on the decoded transmitted intra frames, predicted frames, and bi-directionally predicted frames such that the resulting frame sequence has a desired frame rate.

112. Computer executable logic for decoding video transmitted in a compressed format, the compressed video comprising a sequence of frames that comprise intra frames that do not rely on another frame to encode an image for that frame, predicted frames that rely on a preceding intra frame or predicted frame, bi-directionally predicted frames which rely on a preceding intra frame or predicted frame and/or a subsequent intra frame or predicted frame to encode an image for that frame, and super bi-directional predicted frames which rely on a preceding intra frame, predicted frame, or bi-directional frame and/or a subsequent intra frame, predicted frame, or bi-directional frame to encode an image for that frame, the method comprising:

logic for determining whether super bi-directionally predicted frames are being transmitted;

logic which, if super bi-directionally predicted frames are being transmitted, decodes the super bi-directionally predicted frames; and

logic which, if super bi-directionally predicted frames are not being transmitted, decodes transmitted intra frames, predicted frames, and bi-directionally predicted frames, and produces additional frames based on the decoded transmitted intra frames, predicted frames, and bi-directionally predicted frames such that the resulting frame sequence has a desired frame rate.

113. A hierarchical computer executed method for use in a video compression system for determining motion vectors associated with a block for use in coding decisions, the method comprising:

selecting a first block;

selecting a second block which is a down-sampled block of the first block;

determining multiple motion vectors for the second block; and

refining motion vectors for the first block based, at least in part, on the motion vectors of the second block to produce a set of refined motion vectors for the first block to use in the encoding process.

114. A hierarchical computer executed method according to claim 113 wherein refining the motion vectors is performed by a fractional pixel refinement process.

115. A hierarchical computer executed method according to claim 113 wherein the second block has a size selected from the group consisting of 2x2, 4x4, and 8x8 and the first block has a size selected from the group consisting of 4x4, 8x8 and 16x16.

116. A hierarchical computer executed method for use in a video compression system for determining motion vectors associated with a block for use in coding decisions, the method comprising:

selecting a first block;

selecting a second block which is a down-sampled block of the first block;

selecting a third block which is a down-sampled block of the second block;

determining multiple motion vectors for the third block;

refining motion vectors for the second block based, at least in part, on the motion vectors of the third block to produce a set of refined motion vectors for the second block; and

refining motion vectors for the first block based, at least in part, on the motion vectors of the second block to produce a set of refined motion vectors for the first block to use in the encoding process.

117. A hierarchical computer executed method according to claim 116 wherein the third block has a size selected from the group consisting of 2x2, 4x4, and 8x8, the second block has a size selected from the group consisting of 4x4, 8x8 and the first block has a size selected from the group consisting of 8x8 and 16x16.

118. A hierarchical computer executed method for use in a video compression system for determining motion vectors associated with a block for use in coding decisions, the method comprising:

selecting a first block;

selecting a second block which is a down-sampled block of the first block;

selecting a third block which is a down-sampled block of the second block;

selecting a fourth block which is a down-sampled block of the third block;

determining multiple motion vectors for the fourth block;

refining motion vectors for the third block based, at least in part, on the motion vectors of the fourth block to produce a set of refined motion vectors for the third block;

refining motion vectors for the second block based, at least in part, on the motion vectors of the third block to produce a set of refined motion vectors for the second block; and

refining motion vectors for the first block based, at least in part, on the motion vectors of the second block to produce a set of refined motion vectors for the first block to use in the encoding process.

119. A hierarchical computer executed method according to claim 118 wherein the fourth block has a 2x2 size, the third block has a 4x4 size, the second block has a 8x8 size, and the first block has a 16x16 size.

120. A hierarchical computer executed method according to any of the above claims wherein the multiple motion vectors comprise at least 2, 3, 4 or more candidate motion vectors per block.

121. A method for refining motion vectors for blocks for a given frame, the method comprising:

a) taking a motion vector for a given block and a set of motion vectors for blocks neighboring the given block;

b) computing SAD criteria for the given block using the motion vector for the given block and the set of motion vectors for the neighboring blocks;

c) selecting which of the motion vector for the given block and the set of motion vectors for the neighboring blocks has the smallest SAD criteria for the given block; and

d) if the motion vector with the smallest SAD criteria is not the motion vector for the given block, replacing the motion vector for the given block with the motion vector with the smallest SAD

wherein steps a) – d) are repeated for all blocks in the given frame until no block's motion vector is replaced by a neighboring block's motion vector.

122. A method for refining motion vectors for blocks for a given frame, the method comprising:

a) taking a motion vector for a given block and a set of motion vectors for blocks neighboring the given block;

b) computing SAD criteria for the given block using the motion vector for the given block and the set of motion vectors for the neighboring blocks;

c) selecting which of the motion vector for the given block and the set of motion vectors for the neighboring blocks has the smallest SAD criteria for the given block; and

d) if the motion vector with the smallest SAD criteria is not the motion vector for the given block, replacing the motion vector for the given block with the motion vector with the smallest SAD

wherein steps a) – d) are repeated for multiple iterations.

123. A method according to claims 121 or 122 wherein the given block is a macroblock.

124. A method according to claims 121 or 122 wherein the given block has a size selected from the group consisting of 4x4, 4x8, 8x4, 8x8, 16x8, 8x16, and 16x16.

125. A local motion estimation method comprising:

estimating one or more motion vectors for a macroblock;

subdividing the macroblock into a first set of blocks where each block in the set is smaller than the macroblock;

using the one or more motion vectors for a macroblock to estimate one or more motion vectors for each of the blocks in the first set;

subdividing each of the blocks in the first set into second sets of blocks where each block in the second sets of blocks are smaller than the blocks in the first set of blocks; and

using the one or more motion vectors for each of the blocks in the first set to estimate one or more motion vectors for each of the blocks in the second set of blocks.

126. A local motion estimation method according to claim 125 wherein at least 2, 3, 4 or more motion vectors are estimated for each block or macroblock.

127. A local motion estimation method according to claim 125 wherein the first set of blocks which the macroblock is subdivided into have the same size.

128. A local motion estimation method according to claim 125 wherein the first set of blocks which the macroblock is subdivided into are rectangular.

129. A local motion estimation method according to claim 125 wherein the first set of blocks which the macroblock is subdivided into are square.

130. A local motion estimation method according to claim 125 wherein estimating the motion vectors are performed by searching around pixel matrices centered upon one of the motion vectors for a block from which the block was subdivided.

131. A local motion estimation method according to claim 125 wherein estimating the motion vectors are performed by searching using a SAD cost function.

132. A method for performing motion estimation with non-integer pixel precision comprising:

a) taking a first set of motion vectors mapping a given block to a first reference block;

b) determining a second set of motion vectors by up-sampling the first set of motion vectors and the reference frame to determine a second set of motion vectors which map to a second reference block in the upsampled reference frame; and

c) repeating steps a) and b) where the motion vectors of step b) are employed as the motion vectors taken in step a) until an $1/8^{th}$ pixel precision level is reached, after which, the resulting determined motion vectors are employed in an encoding process.

133. A method according to claim 132 wherein determining the second set of motion vectors is performed by performing a local search around the upsampled first set of motion vectors.

134. A method according to claim 132 wherein the first set of motion vectors comprises at least 2, 3, 4 or more candidate motion vectors per block

135. A method for predicting local motion vectors for a frame being encoded, the method comprising:

a) taking a motion vector for a given block of a frame to be encoded which maps a motion of the given block relative to a reference frame;

b) taking a set of candidate motion vectors which map motion of blocks neighboring the given block relative to one or more reference frames;

c) identifying members of the set of candidate motion vectors which are not validated based on one or more validation rules;

d) compensating for any identified non-validated candidate motion vectors;

e) scaling the validated and compensated motion vectors with respect to the frame being encoded; and

f) computing a predictor motion vector for the motion vector for the given block based on the scaled motion vectors.

136. A method according to claim 135, the method further comprising encoding an error determined by comparing the predictor motion vector and the motion vector for the given block.

137. A method according to claim 135, wherein the candidate motion vectors comprise motion vectors for macroblocks in the frame being encoded and/or blocks within the macroblocks.

138. A method according to claim 135, wherein four of the candidate motion vectors for the given block are selected based on a

position of the given block within a macroblock within the frame being encoded.

139. A method according to claim 135, wherein one of the candidate motion vectors in the set is a motion vector used by a block in a reference frame.

140. A method according to claim 135, wherein four of the candidate motion vectors for the given block are selected based on a position of the given block within a macroblock within the frame being encoded, and a fifth of the candidate motion vectors in the set is a motion vector used by a block in a reference frame.

141. A method according to claim 135, wherein one of the candidate motion vectors in the set is a motion vector used by a block in a reference frame that is in a same position in the reference frame as the given block in the frame being encoded.

142. A method according to claim 135, wherein four of the candidate motion vectors for the given block are selected based on a position of the given block within a macroblock within the frame being encoded, and a fifth of the candidate motion vectors in the set is a motion vector used by a block in a reference frame that is in a same position in the reference frame as the given block in the frame being encoded.

143. A method according to claim 135, at least one candidate motion vector is identified as being invalid.

144. A method according to claim 135, wherein scaling of motion vectors is performed by the formula

$$ScaledMVi = ((t0-t1)/(t1-t2)) * MVi$$

where

Mvi is a candidate motion vector predictor,

ScaledMVi is the scaled candidate motion vector predictor,

t0 is time of current motion vector,

t1 is the time of the frame to which the current motion vector references, and

t2 is the time of the frame, to which a motion vector in the co-located block of the frame at time t1, references.

145. A method according to claim 135, wherein one or more validation rules are selected from the group consisting of:

if any candidate motion vector is not coded, it is invalid;

if only one candidate motion vector is invalid, it is set to zero;

if two candidate motion vectors are invalid, the two motion vectors are discarded;

if three candidate motion vectors are invalid, two are discarded and a third is set to zero;

if four candidate motion vectors are invalid, they are each set to a fifth motion vector candidate; and

if five candidate motion vectors are invalid, they are each set to zero.

146. A method for predicting local motion vectors for a frame being encoded, the method comprising:

a) taking a motion vector for a given block of a frame to be encoded which maps a motion of the given block relative to a reference frame;

b) taking candidate motion vectors which map motion of blocks neighboring the given block relative to one or more reference frames;

c) identifying members of the candidate motion vectors which are not validated based on one or more validation rules;

d) compensating for any identified non-validated candidate motion vectors;

e) scaling the validated and compensated motion vectors with respect to the frame being encoded;

f) identifying an additional candidate motion vector by taking an average of the scaled motion vectors and identifying a motion vector for a block in a reference frame to which the average motion vector points;

g) identifying whether the additional candidate motion vector is valid based on one or more validation rules and compensating for the additional candidate motion vector if it is invalid;

h) scaling the additional candidate motion vector with respect to the frame being encoded; and

i) computing a predictor motion vector for the motion vector for the given block based on the scaled candidate motion vectors.

147. A method for estimating an affine model, the method comprising:

a) taking an initial set of motion vectors which map macroblocks in a frame to be encoded to corresponding macroblocks in a reference frame;

b) determining a set of affine equations based on the initial set of motion vectors;

c) determining an affine model by solving the determined set of affine equations;

d) determining a modified set of motion vectors based on the determined affine model;

e) eliminating motion vectors from the modified set of motion vectors which are inconsistent with the initial motion vector; and

f) determining a final affine model by repeating steps b-e where the motion vectors determined in step d which are not eliminated in step

e are used as the initial set of motion vectors in step b until either (i) a predetermined number of iterations have been performed, or (ii) a predetermined accuracy threshold for the modified motion vectors has been reached.

148. A method according to claim 147 wherein the set of affine equations comprises at least six equations.

149. A method according to claim 147 wherein the set of affine equations exceeds a number of affine model parameters used to determine the set of affine equations.

150. A method according to claim 147 wherein initial values for the affine parameters are determined using a least squares estimation.

151. A method according to claim 147 wherein the affine model is expressed by the equation

$$u_k = a_1 x_k + b_1 y_k + c_1, \text{ and}$$

$$v_k = a_2 x_k + b_2 y_k + c_2$$

where u_k and v_k are predicted x and y components of a motion vector corresponding to a macroblock k in the frame being encoded and variables a_1, b_1, c_1, a_2, b_2 and c_2 are affine model parameters to be determined.

152. A method according to claim 151 wherein values for u_k and v_k are determined based on motion vectors derived during local motion estimation.

153. A method according to claim 147 wherein determining the modified set of motion vectors based on the determined set of affine

equations is performed by using the affine model to construct a predicted frame on a pixel-by-pixel basis.

154. A method according to claim 147 wherein eliminating motion vectors from the modified set of motion vectors which are inconsistent with the affine model is performed by selecting a filtering threshold defining a maximum allowed deviation from a corresponding motion vector derived during local motion estimation, and eliminating those motion vectors which do not satisfy the threshold.

155. A method according to claim 147 wherein steps a-d are repeated for at least 2, 3, 4, 5, 6, 7, 8, 9 or more iterations.

156. A method according to claim 147 wherein steps a-d are repeated until the set of motion vectors determined in step c and not eliminated in step d satisfy a final accuracy threshold.

157. A method according to claim 147 wherein the accuracy thresholds employed in the multiple iterations of steps a-d are pre-defined for each iteration.

158. A method according to claim 147 wherein the accuracy threshold decreases with each iteration.

159. A method according to claim 147 wherein the accuracy threshold comprises a magnitude threshold and a phase threshold.

160. A method according to claim 147 the final accuracy threshold is 0.5 pixels for a magnitude of the motion vectors and 5 degrees for the phase of the motion vectors.

161. A method for encoding a bidirectionally predicted frame comprising:

computing an affine model for a given bi-directionally predicted frame to be encoded where a preceding frame is used as a reference frame in the affine model;

warping the preceding reference frame;

determining a residue between the given bi-directionally predicted frame to be encoded and the warped preceding reference frame;

encoding the given bi-directionally predicted frame to be encoded with reference to the residue.

162. A method according to claim 161 wherein the affine model is computed by

a) taking an initial set of motion vectors which map macroblocks in a frame to be encoded to corresponding macroblocks in a reference frame;

b) determining a set of affine equations based on the initial set of motion vectors;

c) determining an affine model by solving the determined set of affine equations;

d) determining a modified set of motion vectors based on the determined affine model;

e) eliminating motion vectors from the modified set of motion vectors which are inconsistent with the initial motion vector; and

f) determining a final affine model by repeating steps b-e where the motion vectors determined in step d which are not eliminated in step e are used as the initial set of motion vectors in step b until either (i) a predetermined number of iterations have been performed, or (ii) a predetermined accuracy threshold for the modified motion vectors has been reached.

163. A method for encoding a bidirectionally predicted frame comprising:

computing an affine model for a given bi-directionally predicted frame to be encoded where a subsequent frame is used as a reference frame in the affine model;

warping the subsequent reference frame;

determining a residue between the given bi-directionally predicted frame to be encoded and the warped subsequent reference frame;

encoding the given bi-directionally predicted frame to be encoded with reference to the residue.

164. A method according to claim 163 wherein the affine model is computed by

a) taking an initial set of motion vectors which map macroblocks in a frame to be encoded to corresponding macroblocks in a reference frame;

b) determining a set of affine equations based on the initial set of motion vectors;

c) determining an affine model by solving the determined set of affine equations;

d) determining a modified set of motion vectors based on the determined affine model;

e) eliminating motion vectors from the modified set of motion vectors which are inconsistent with the initial motion vector; and

f) determining a final affine model by repeating steps b-e where the motion vectors determined in step d which are not eliminated in step e are used as the initial set of motion vectors in step b until either (i) a predetermined number of iterations have been performed, or (ii) a predetermined accuracy threshold for the modified motion vectors has been reached.

165. A method for refining a global motion estimation for a given block of a current frame being encoded, the method comprising:

computing an affine model for the current frame;

warping a reference frame for the current frame;

determining a prediction error between the given block in the current frame and a block within the warped reference frame;

determining motion vectors between the given block and the block within the warped reference frame; and

modifying the prediction error based on the determined motion vectors.

166. A method according to claim 165 wherein the affine model is computed by

a) taking an initial set of motion vectors which map macroblocks in a frame to be encoded to corresponding macroblocks in a reference frame;

b) determining a set of affine equations based on the initial set of motion vectors;

c) determining an affine model by solving the determined set of affine equations;

d) determining a modified set of motion vectors based on the determined affine model;

e) eliminating motion vectors from the modified set of motion vectors which are inconsistent with the initial motion vector; and

f) determining a final affine model by repeating steps b-e where the motion vectors determined in step d which are not eliminated in step e are used as the initial set of motion vectors in step b until either (i) a predetermined number of iterations have been performed, or (ii) a predetermined accuracy threshold for the modified motion vectors has been reached.

167. A method for encoding a given block employing multiple hypotheses, the method comprising:

taking multiple reference blocks from multiple reference frames; and

encoding the given block based on a combination of the multiple reference blocks.

168. A method according to claim 167 where taking multiple reference macroblocks comprises selecting the multiple reference blocks from a larger set of reference blocks by taking groups of reference blocks of the larger set of reference macroblocks and selecting a subset of those groups based on a cost function.

169. A method according to claim 168 where the cost function for each group of reference blocks is based on a cost of encoding the given macroblock with respect to a combination of reference blocks.

170. A method according to claim 168 wherein two or more of the groups of reference blocks comprise a same block.

171. A method according to claim 167 wherein the combination of the multiple reference blocks are combined to comprise a predictor block for the block being encoded.

172. A method according to claim 167 wherein the groups of reference blocks comprise at least two, three, four or more blocks.

173. A method according to claim 167 wherein the given block is a macroblock.

174. A method according to claim 167 wherein the reference block is a macroblock.

175. A method according to claim 167 wherein the given block has a size selected from the group consisting of 4x4, 4x8, 8x4, 8x8, 16x8, 8x16, and 16x16.

176. A method according to claim 167 wherein the reference block has a size selected from the group consisting of 4x4, 4x8, 8x4, 8x8, 16x8, 8x16, and 16x16.

177. A method for encoding a given block of a macroblock, the method comprising:

encoding the given block on a pixel by pixel basis using a predictor pixel value obtained by combining reference pixel values from multiple reference frames as indicated by motion vectors for two blocks neighboring the given block and a motion vector for the given block.

178. A method according to claim 177, wherein the motion vectors for two blocks neighboring the given blocks have been modified by translation of their coordinates from their original coordinates to coordinates of a current pixel being encoded.

179. A method according to claim 177 or 178 wherein combining reference pixel values comprises weighing the reference pixel values from the multiple reference frames according to the position of the current pixel being encoded in the given block.

180. A method according to claim 177 wherein the pixel being encoded in the given block is encoded with respect to the predictor value obtained by weighing the reference pixel values from the multiple reference frames according to the position of the current pixel.

181. A method for deciding a coding mode for a given block based on a minimum rate-distortion cost function, the method comprising:

taking multiple reference blocks from multiple reference frames;

taking multiple coding modes for the given block with respect to a reference block or a combination of reference blocks; and

determining a coding decision for the given block by minimizing a cost as determined by a rate-distortion cost function.

182. A method according to claim 181 wherein determining the coding decision comprises selecting a reference block or a combination of reference blocks for a given block, and selecting a block coding mode with respect to the reference block or a combination of reference blocks based on the rate-distortion cost function.

183. A method according to claim 182 wherein selecting the block coding mode for the given block with respect to a reference block or a combination of reference blocks comprises selecting the coding mode from a set of all possible block coding modes based on the rate-distortion cost function.

184. A method according to claims 181, 182, or 183 wherein the rate-distortion cost function employs a weighted combination of a distortion cost function and a rate cost function for a given block with respect to a selected reference block or a combination of reference blocks and the selected block coding mode.

185. A method according to claim 184 wherein the rate-distortion cost function is determined by

taking transformed coefficients of the current block;

taking a scan order for grouping the transformed coefficients of the current block;

grouping one or more transformed coefficient to form a set of coefficients;

modifying attributes of each of the coefficients in the set based on a rate-distortion cost function for the set of the coefficients;

repeating the steps of grouping and modifying iteratively until all of the transformed coefficients are processed;

calculating the distortion cost function for the given block based on the modified transformed coefficients of the given block; and

calculating the rate cost function for the given block based on the modified transformed coefficients of the given block.

186. A method according to claim 185 wherein the rate-distortion cost function for the set of the coefficients comprises a weighted combination of the distortion cost function and the rate cost function of the given set of transformed coefficients

187. A method according to claim 181 wherein the combination of reference blocks comprises at least two blocks.

188. A method according to claim 181 wherein the given block is a macroblock.

189. A method according to claim 181 wherein the reference block is a macroblock.

190. A method according to claim 181 wherein the given block and the reference block have a size selected from the group consisting of 4x4, 4x8, 8x4, 8x8, 16x8, 8x16, and 16x16.

191. A method for encoding a given macroblock comprising:

a) taking a current macroblock of a current frame;

b) determining a reference macroblock of a different, reference frame for the current macroblock based on motion estimation;

c) selecting for a block from the current macroblock a corresponding block from the reference macroblock based on a block level motion estimation for the current block or based on a corresponding position in the macroblock;

d) sorting pixels for each line of the reference block based on pixel values within the line;

e) identifying a permutation for each line in the sorted reference block corresponding to a modified order of the pixels in the line as a result of the sorting;

f) permuting pixels for each line of the current block based on the corresponding permutation for each line of the corresponding block of the reference macroblock.; and

g) calculating a prediction error block based on a difference between the permuted current block and the sorted reference block;

wherein steps a – g are repeated for all blocks of the current macroblock and a frequency transformation is applied to each of the blocks of the residual macroblock so obtained.

192. A method for decoding a macroblock comprising:

receiving a prediction error block in a bitstream which corresponds to a current block to be decoded;

taking the corresponding reference block from an already decoded reference frame for the current block to be decoded;

sorting pixels for each line of the reference block based on pixel values within the line;

identifying a permutation for each line in the sorted reference block corresponding to a modified order of the pixels in the line as a result of the sorting;

adding the sorted reference block to the corresponding prediction error block to obtain a permuted current block; and

using the identified line permutations to inverse permute the permuted current block lines to obtain the reconstructed current block.

193. A method for encoding a given macroblock comprising:

a) taking a prediction error block from a prediction error macroblock for the given macroblock being encoded;

b) for each line of the prediction error block,

permute the line in all possible combinations to generate all possible permutations,

optimally match the different permutations generated to a target signal from a target signal matrix which comprises targets signals used in transforming the prediction error block,

identify which of the matched permutations has a lowest cost with respect to a target signal based on a cost function,

permute all lines in the prediction error block according to the permutation for the current line identified as having the lowest cost,

optimally match each line of the permuted prediction error block to a target signal from the target signal matrix based on a cost function, and

determine and record a cumulative block level cost by summing the costs for each optimally matched line and associate the determined cumulative block level cost with the current line being processed from the prediction error block;

c) determine a minimum block level cost from the cumulative block level costs determined in step b corresponding to each line from the prediction error block, and record the associated permutation;

d) use the permutation identified in step c to permute all the lines in the prediction error block; and

e) determine and transmit results of a frequency transform of the permuted prediction error block determined in step d.

194. A method for encoding a given macroblock comprising:

a) determining a block for the given macroblock being encoded;

b) for each line of the block,

permute the line in all possible combinations to generate all possible permutations,

optimally match the different permutations generated to a target signal from a target signal matrix which comprises targets signals used in transforming the block,

identify which of the matched permutations has a lowest cost with respect to a target signal based on a cost function,

permute all lines in the block according to the permutation for the current line identified as having the lowest cost,

optimally match each line of the permuted block to a target signal from the target signal matrix based on a cost function, and

determine and record a cumulative block level cost by summing the costs for each optimally matched line and associate the determined cumulative block level cost with the current line being processed from the block;

c) determine a minimum block level cost from the cumulative block level costs determined in step b corresponding to each line from the block, and record the associated permutation;

d) use the permutation identified in step c to permute all the lines in the block; and

e) determine and transmit results of a frequency transform of the permuted block determined in step d.

195. A method for decoding a macroblock comprising:

receiving a permuted prediction error block from a bitstream which corresponds to a current block to be decoded;

applying an inverse permutation which was used to encode the prediction error block to the prediction error block to produce an inverse permuted prediction error block; and

adding the inverse permuted prediction error block to an already decoded reference block to obtain a reconstructed current block.

196. A method for decoding a macroblock comprising:

receiving a permuted block from a bitstream; and

applying an inverse permutation which was used to encode the block to the block to produce an inverse permuted block.

197. A method for reducing block artifacts in decoded images comprising:

taking a decoded block having edge pixels defining a perimeter of the block; and

for each edge pixel

taking an edge pixel as a current pixel,

determining a local variance for the current pixel by comparing the current pixel to pixels neighboring the current pixel,

determining absolute differences between two neighboring pixels of the current pixel a) normal to the edge, b) in a positive diagonal direction relative to the edge, and c) in a negative diagonal direction relative to the edge,

constructing a pixel-based directional filter based on the determined local variance and the determined absolute differences, and

calculating a new value for the current pixel using the constructed pixel-based directional filter.

198. A method for smoothing artifacts within a decoded image comprising:

constructing a cost function including a closeness term and a smoothness term;

determining a system of linear equations by minimizing the cost function; and

iteratively solving the equations for smoothed values for all pixels.

199. A method for reducing contours in a decoded image, the method comprising:

taking a decoded image;

for blocks of the decoded image, computing local variances for each block;

using the computed local variances to determine smooth regions of the decoded image;

segmenting the detected smooth regions;

randomly selecting pixels within each smooth region; and

adding random noise to the selected pixels.

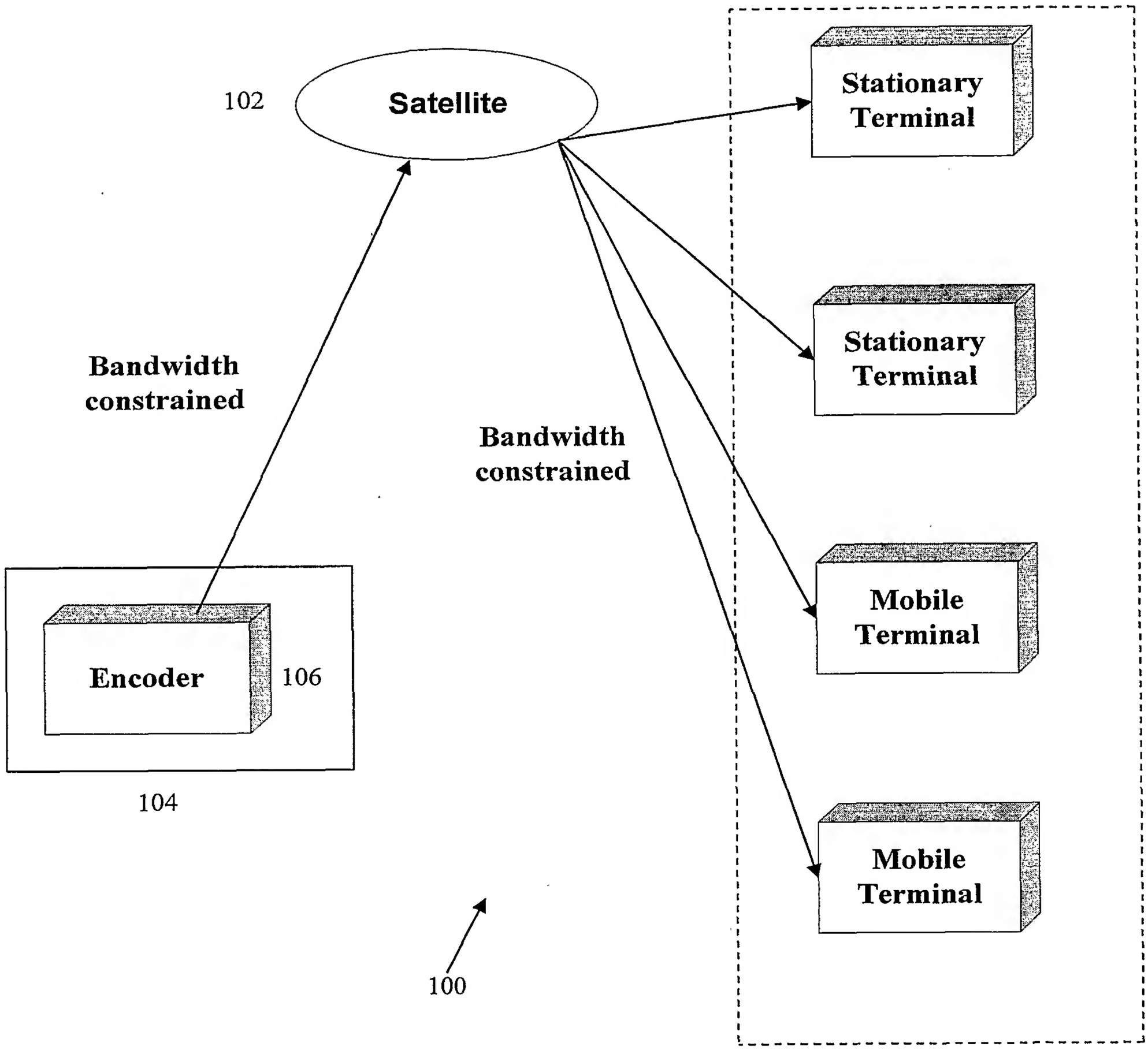

102
Satellite
Stationary Terminal
Stationary Terminal
Mobile Terminal
Mobile Terminal
Bandwidth constrained
Bandwidth constrained
Encoder
106
104
100
108

FIGURE 1A

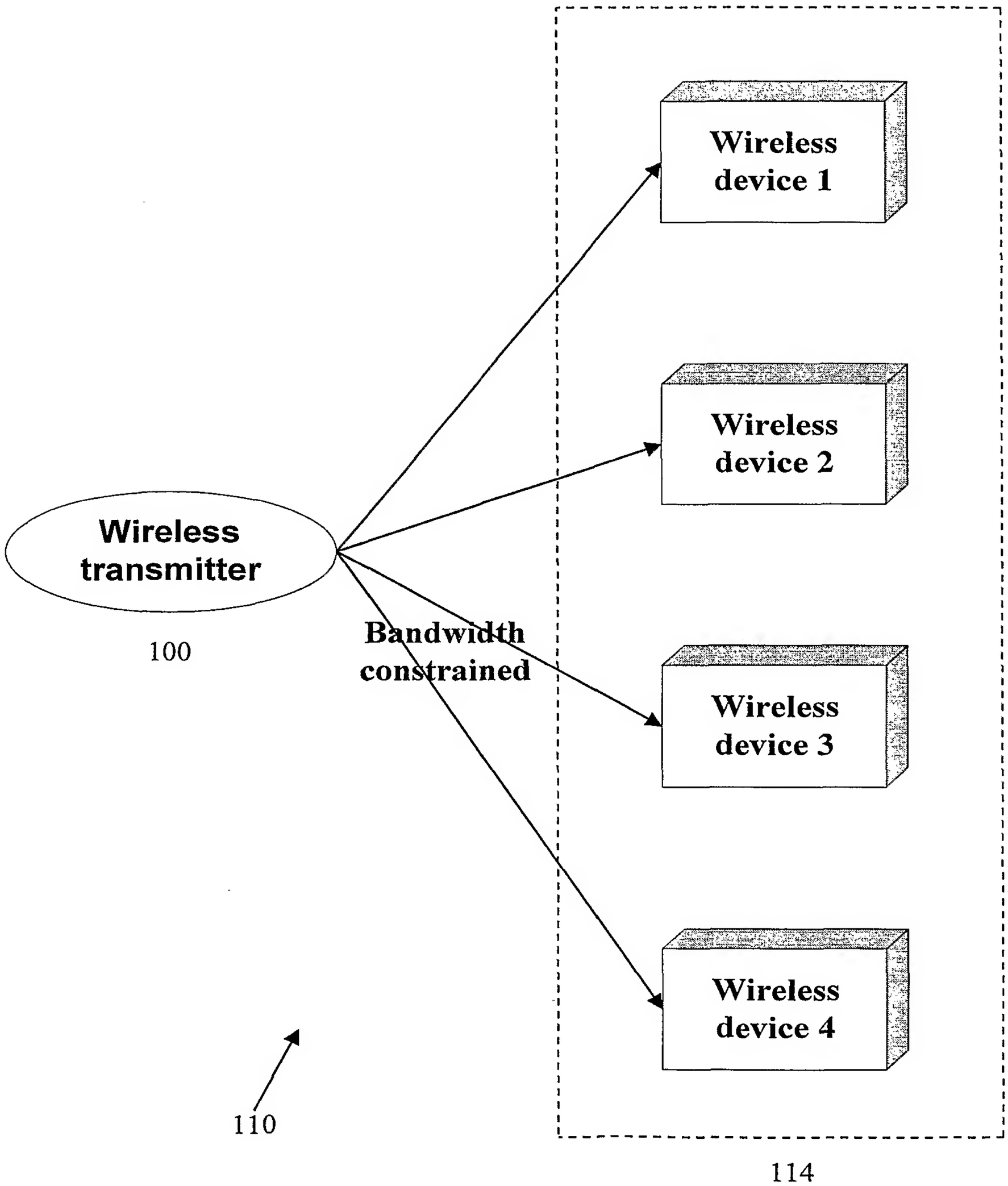

Wireless device 1
Wireless device 2
Wireless device 3
Wireless device 4
Wireless transmitter
100
Bandwidth constrained
110
114

FIGURE 1B

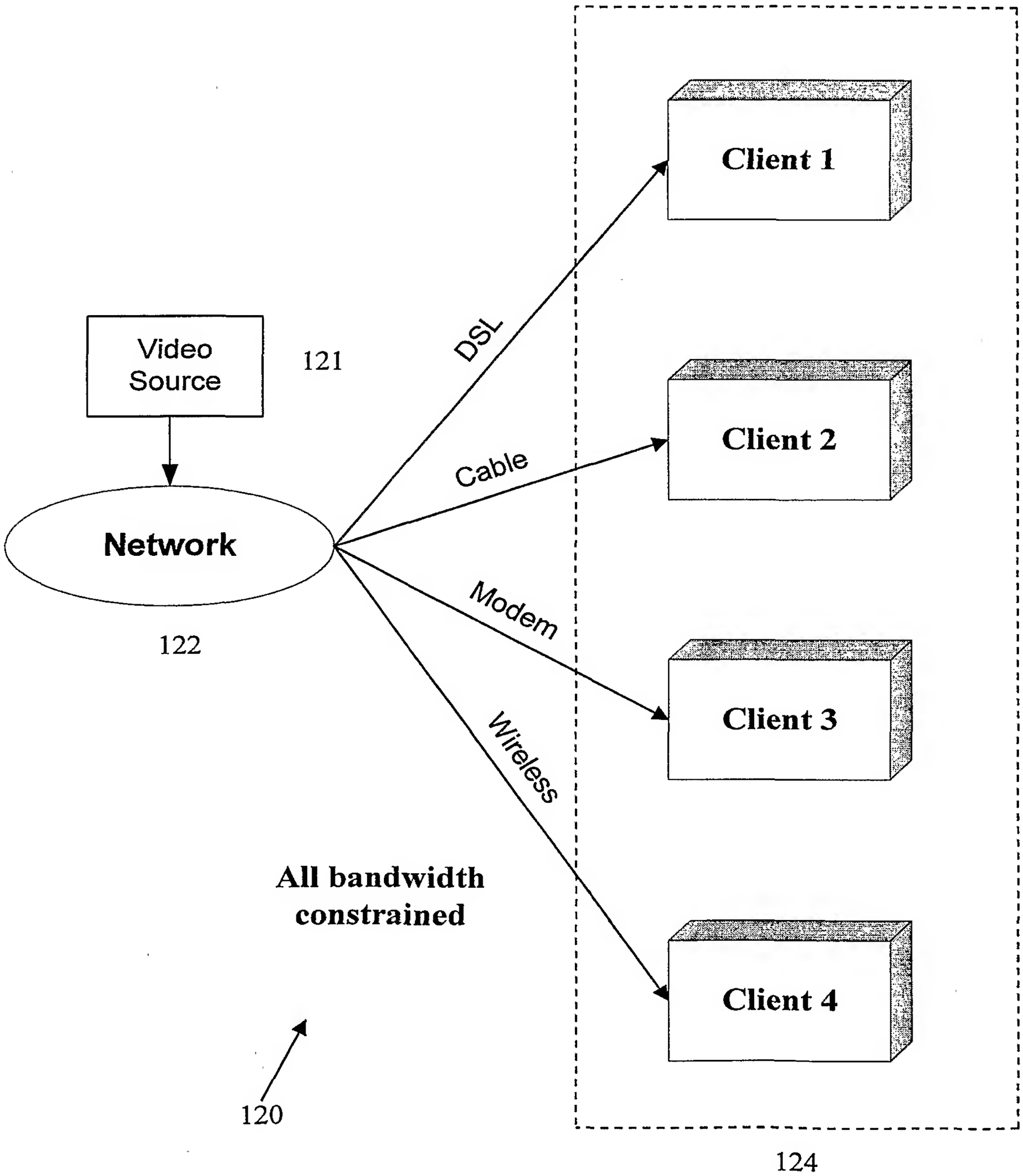
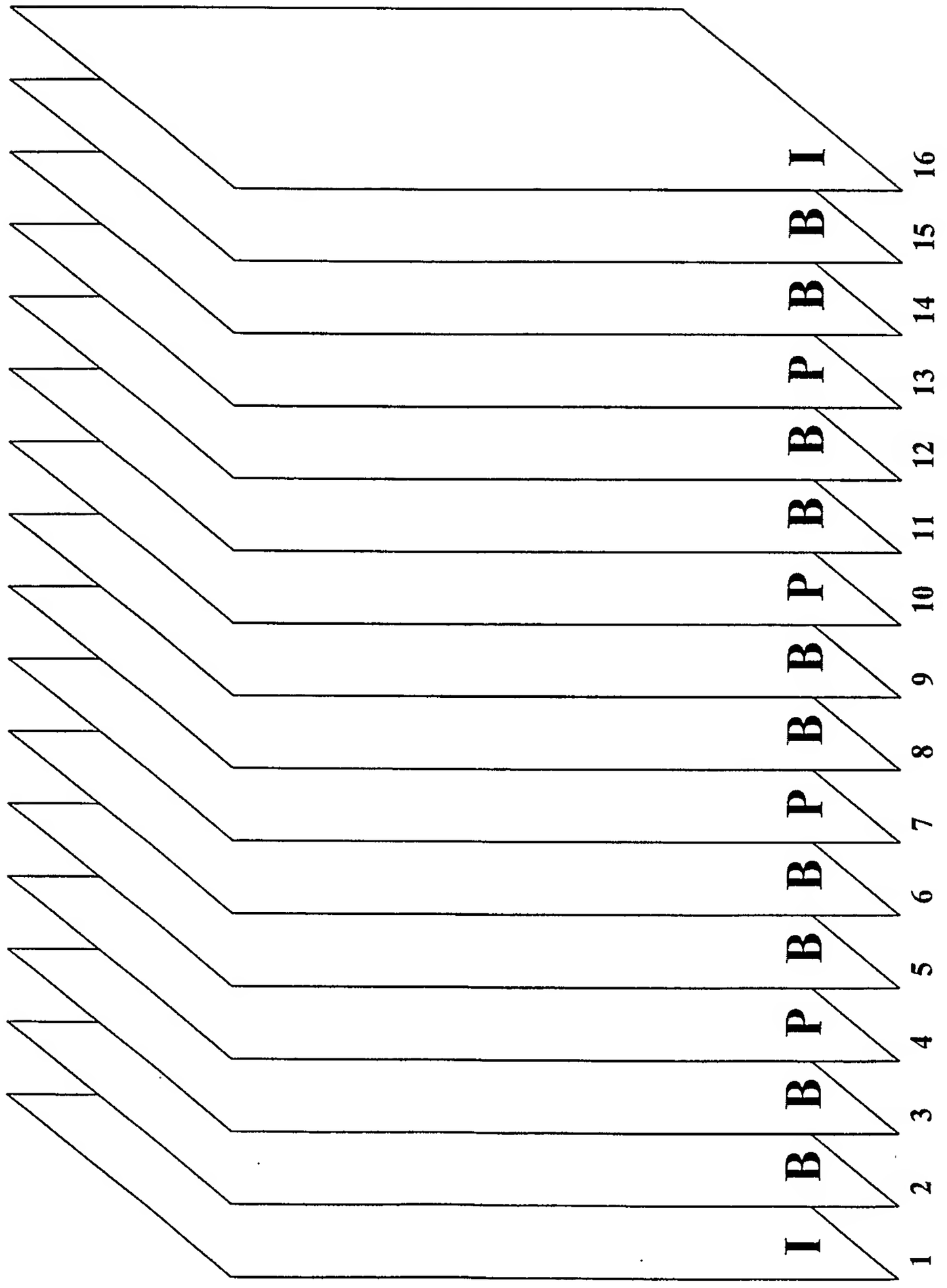

Client 1
Video Source
121
DSL
Client 2
Cable
Network
Modem
Client 3
122
Wireless
All bandwidth constrained
Client 4
120
124

FIGURE 1C



PRIOR ART

FIGURE 2A

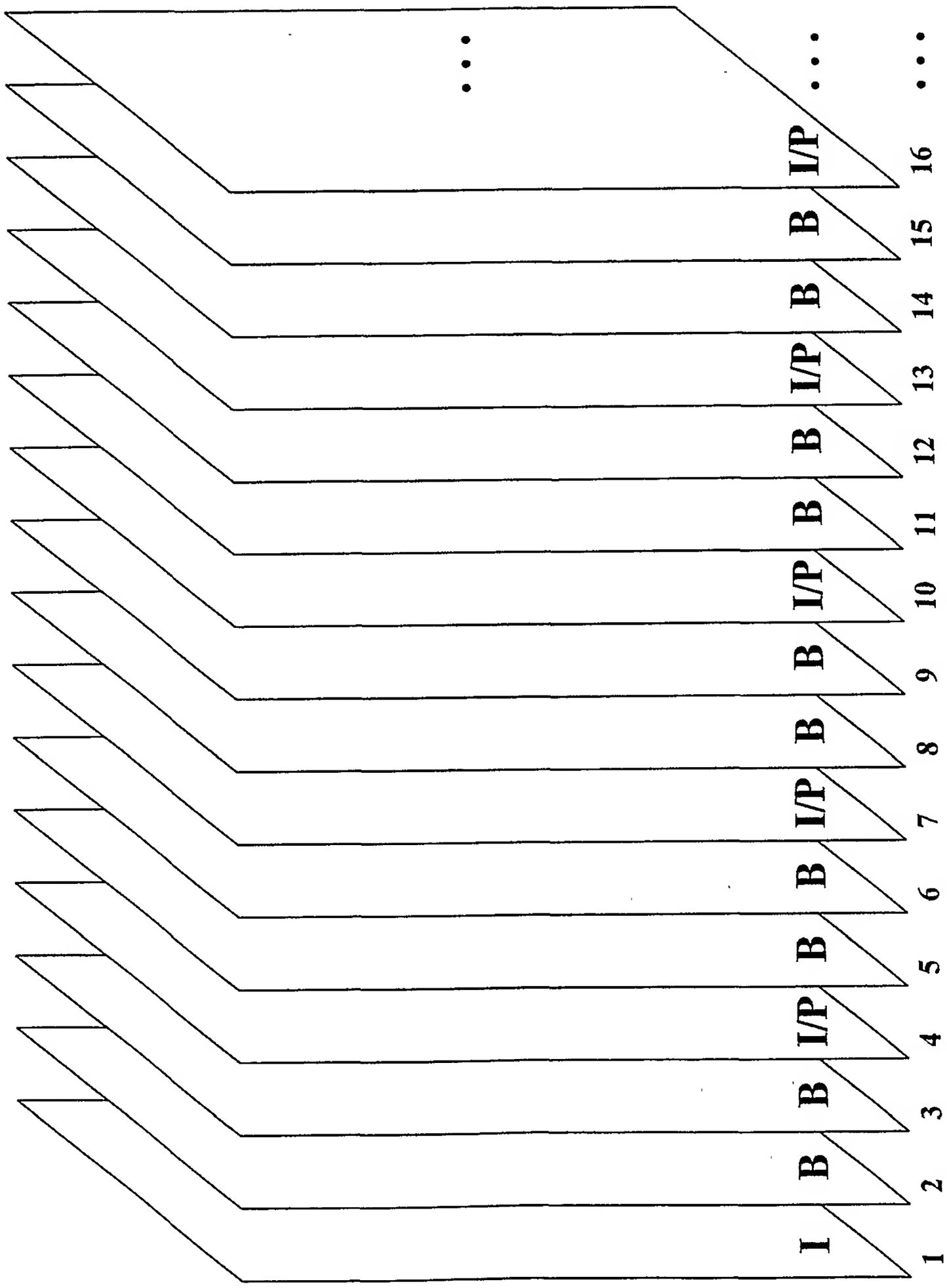

I
B
B
I/P
B
B
I/P
B
B
I/P
B
B
I/P
B
B
I/P
B
B
I/P
B
B
I/P
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

FIGURE 2B

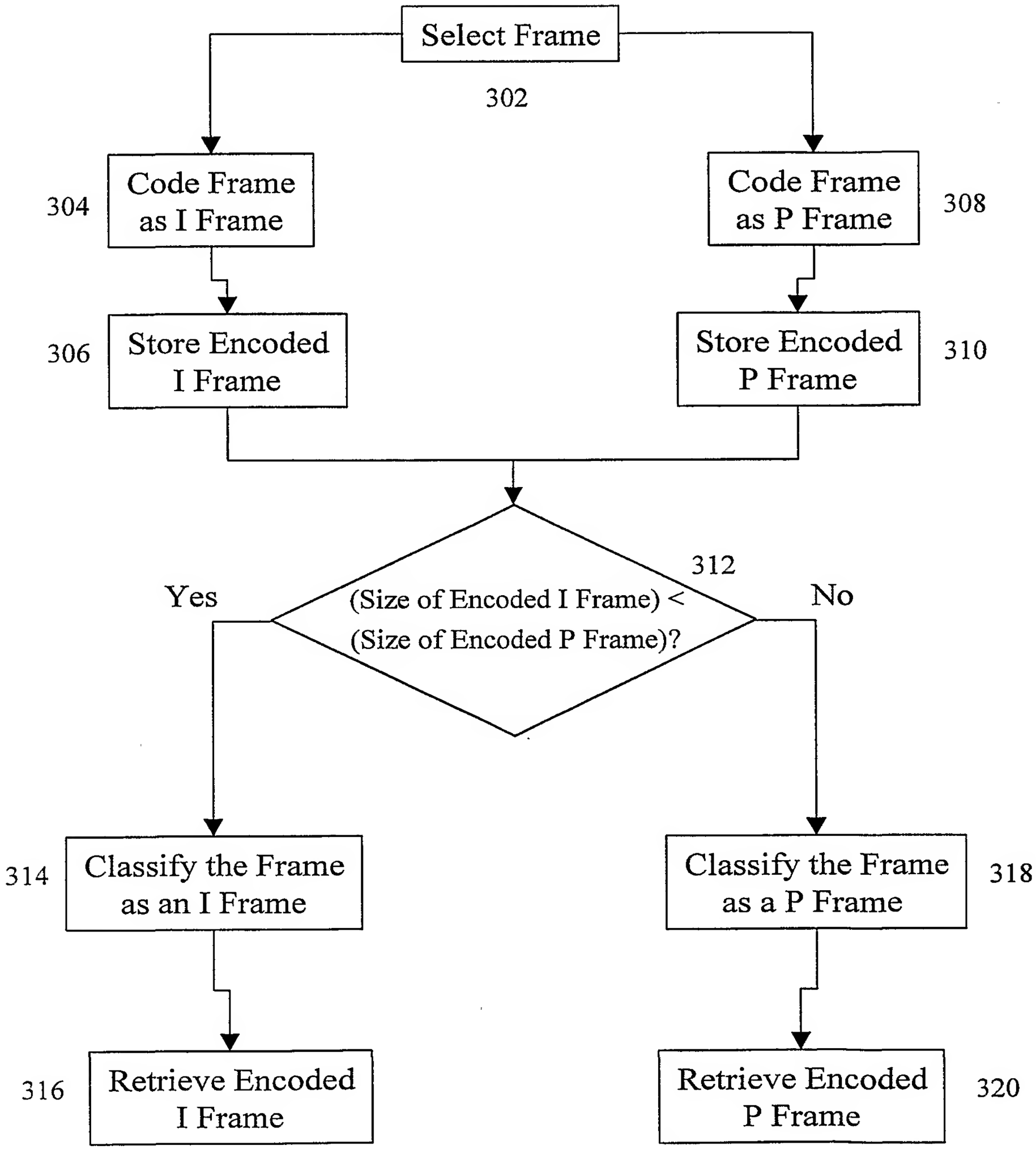

Select Frame
302
304
Code Frame as I Frame
308
Code Frame as P Frame
306
Store Encoded I Frame
310
Store Encoded P Frame
312
Yes
(Size of Encoded I Frame) < (Size of Encoded P Frame)?
No
314
Classify the Frame as an I Frame
318
Classify the Frame as a P Frame
316
Retrieve Encoded I Frame
320
Retrieve Encoded P Frame

FIGURE 3A

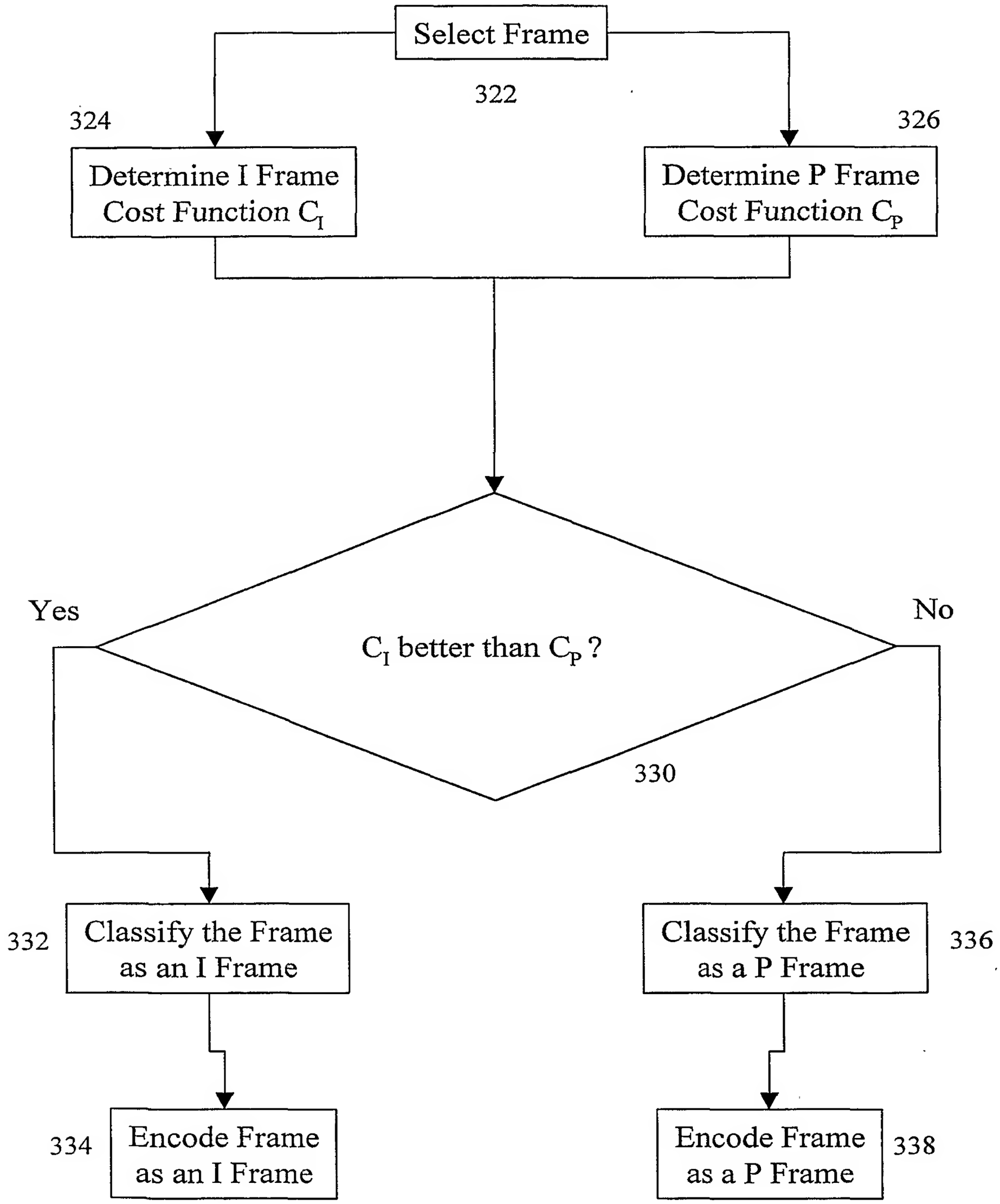

Select Frame
322
324
326
Determine I Frame Cost Function C_I
Determine P Frame Cost Function C_P
Yes
No
C_I better than C_P ?
330
332
Classify the Frame as an I Frame
Classify the Frame as a P Frame
336
334
Encode Frame as an I Frame
Encode Frame as a P Frame
338

FIGURE 3B

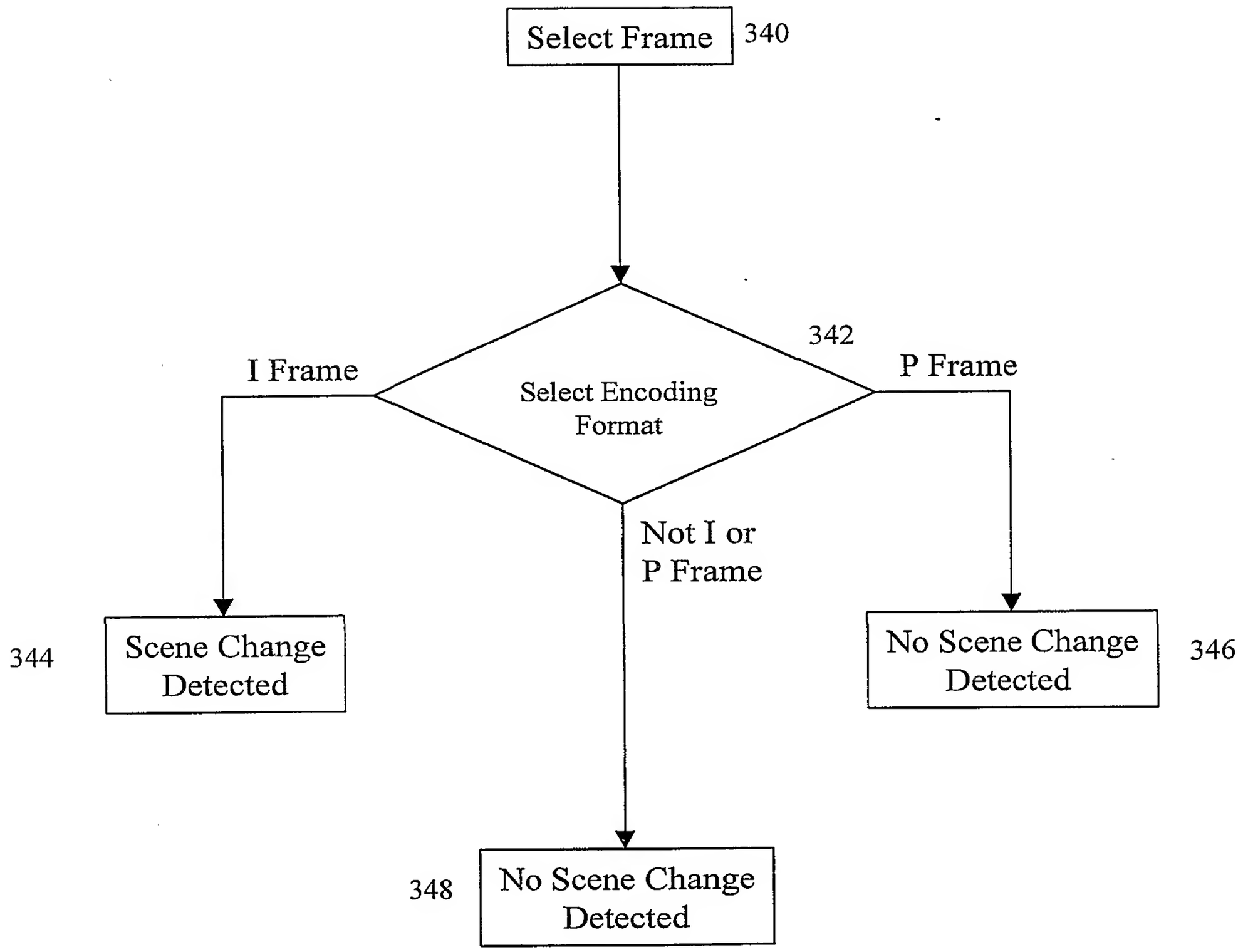

Select Frame
340
342
Select Encoding Format
I Frame
P Frame
Not I or P Frame
344
Scene Change Detected
No Scene Change Detected
346
348
No Scene Change Detected

FIGURE 3C

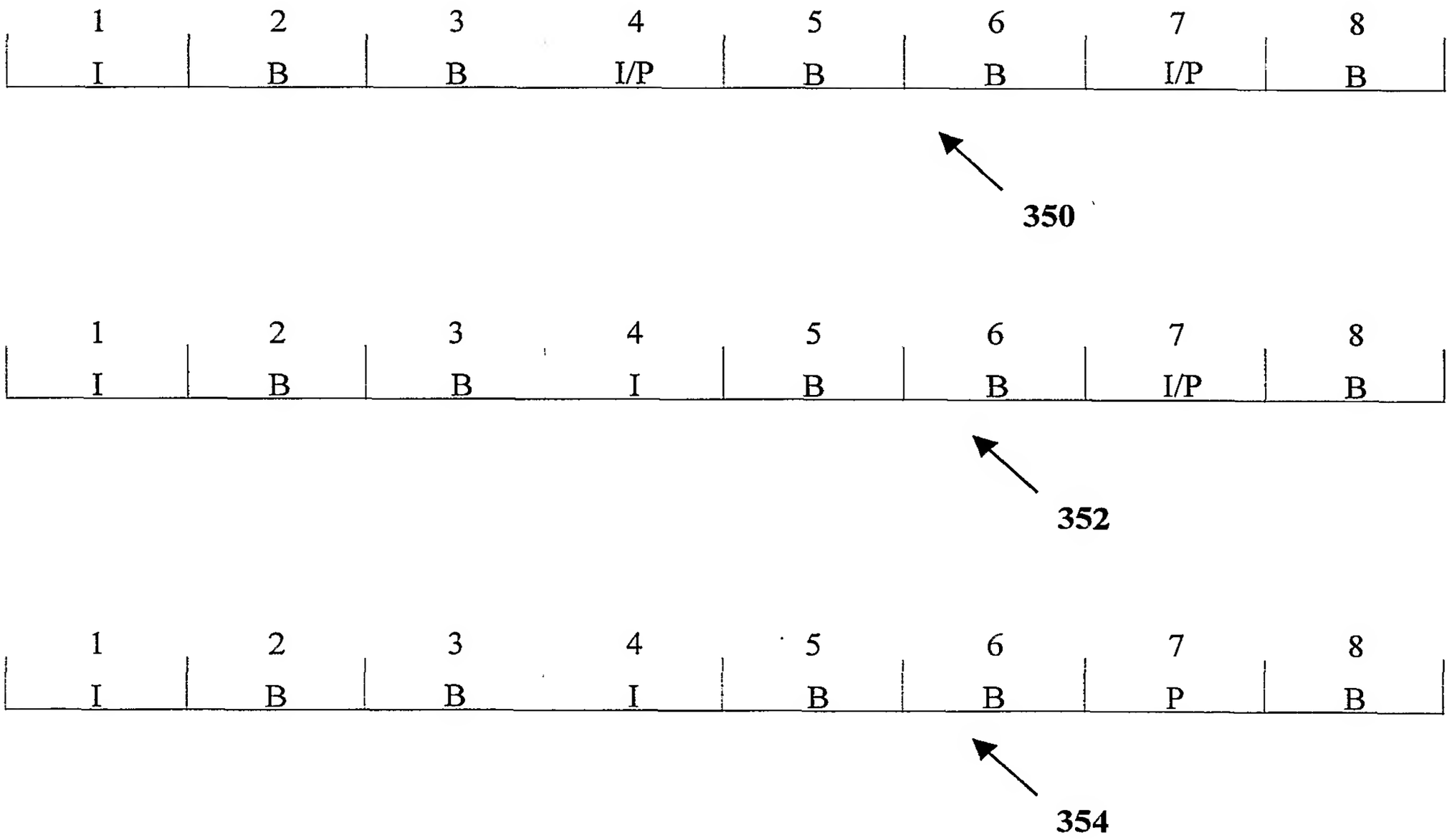

1 2 3 4 5 6 7 8
I B B I/P B B I/P B
350
1 2 3 4 5 6 7 8
I B B I B B I/P B
352
1 2 3 4 5 6 7 8
I B B I B B P B
354

FIGURE 3D

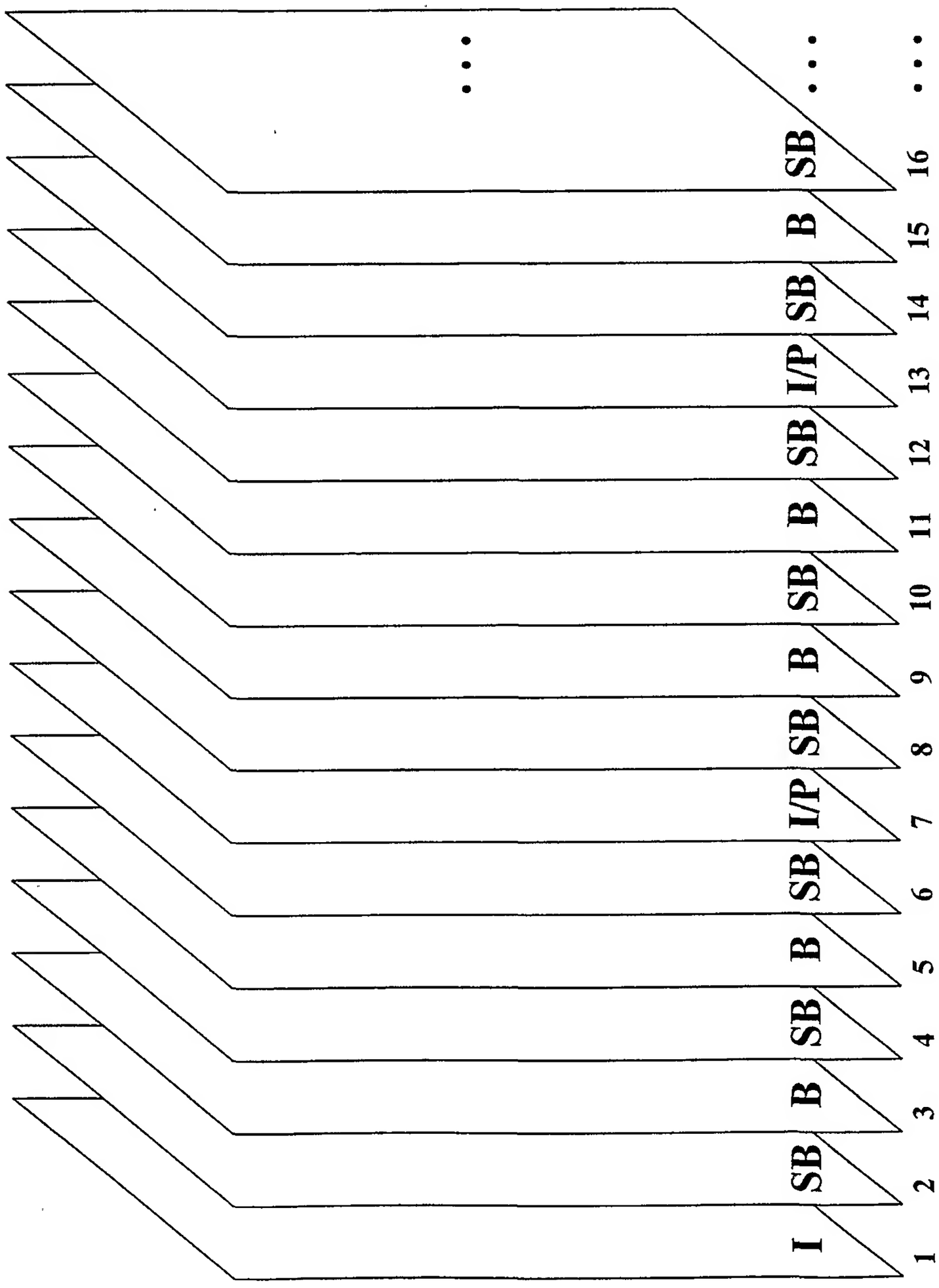

I
SB
B
SB
I/P
B
SB
B
SB
I/P
B
SB
B
SB
I/P
B
SB
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16

FIGURE 3E

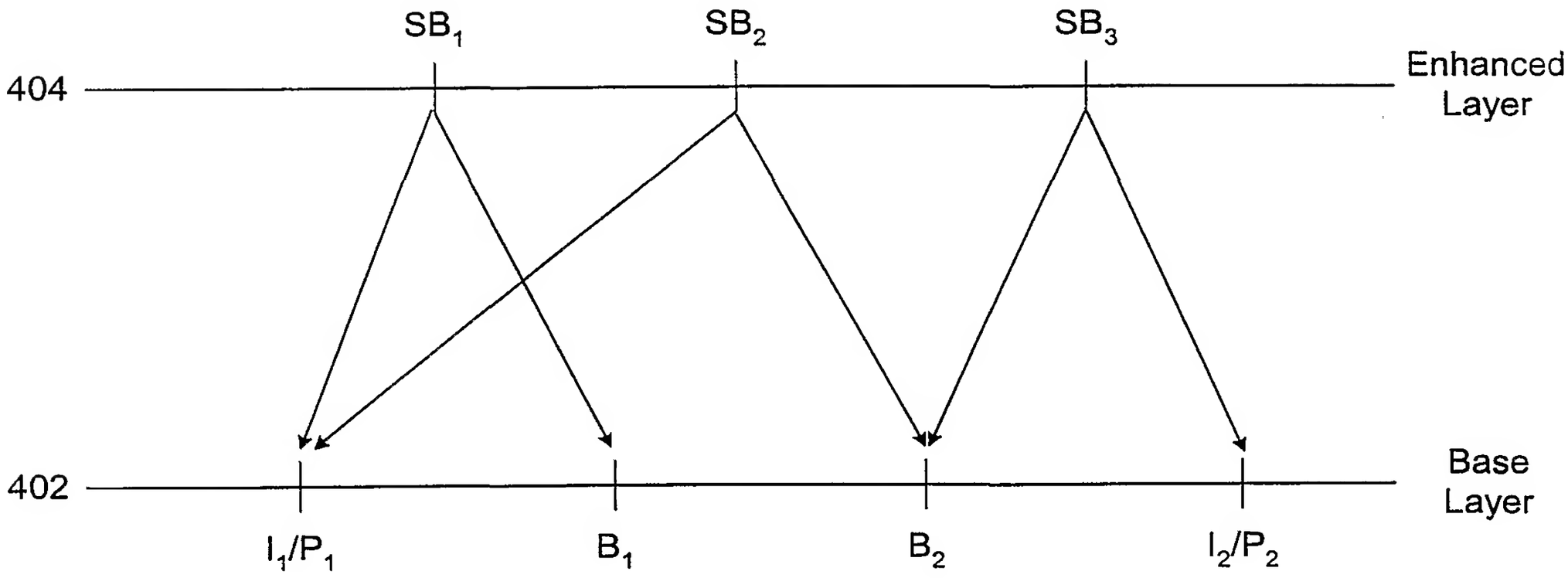


FIGURE 4A

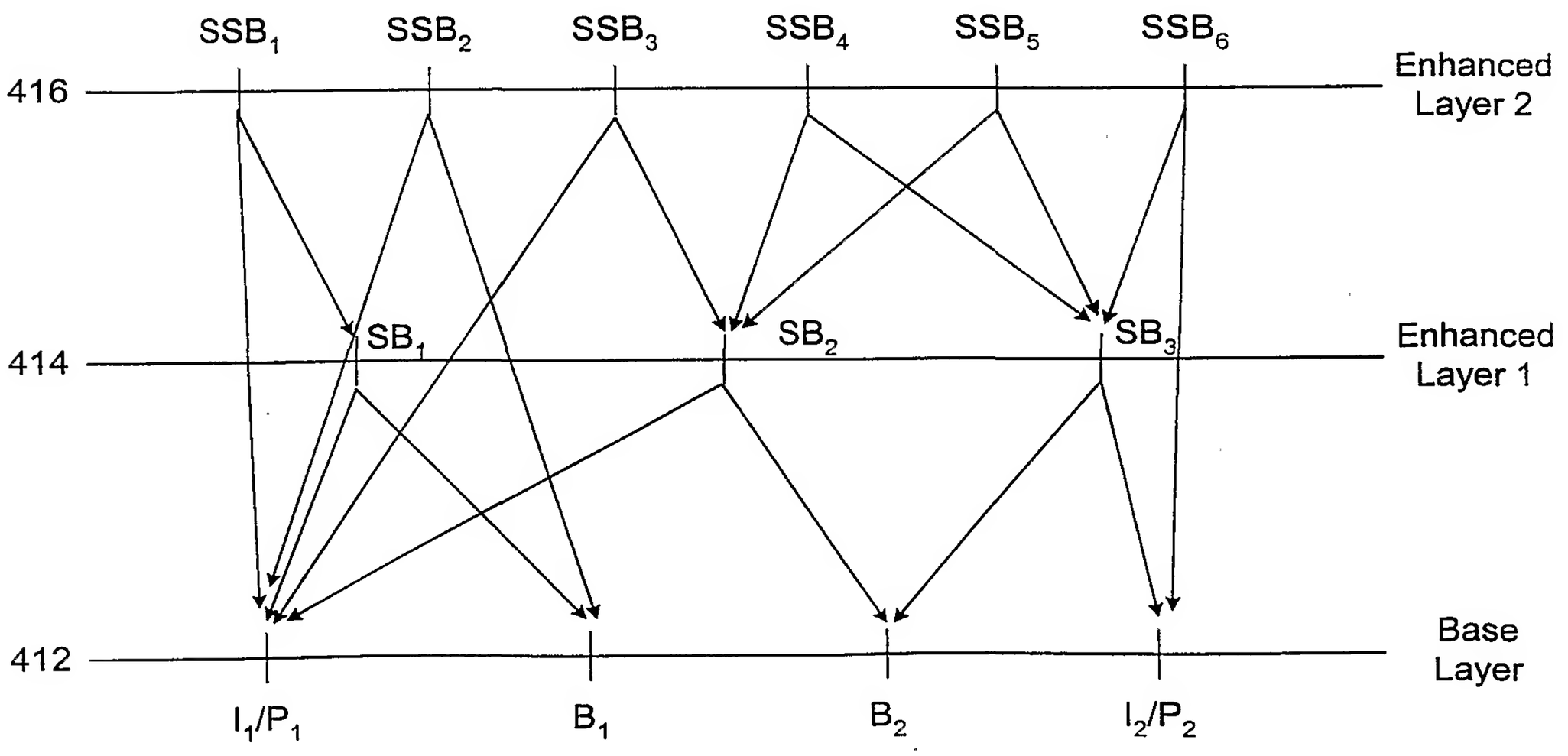


FIGURE 4B

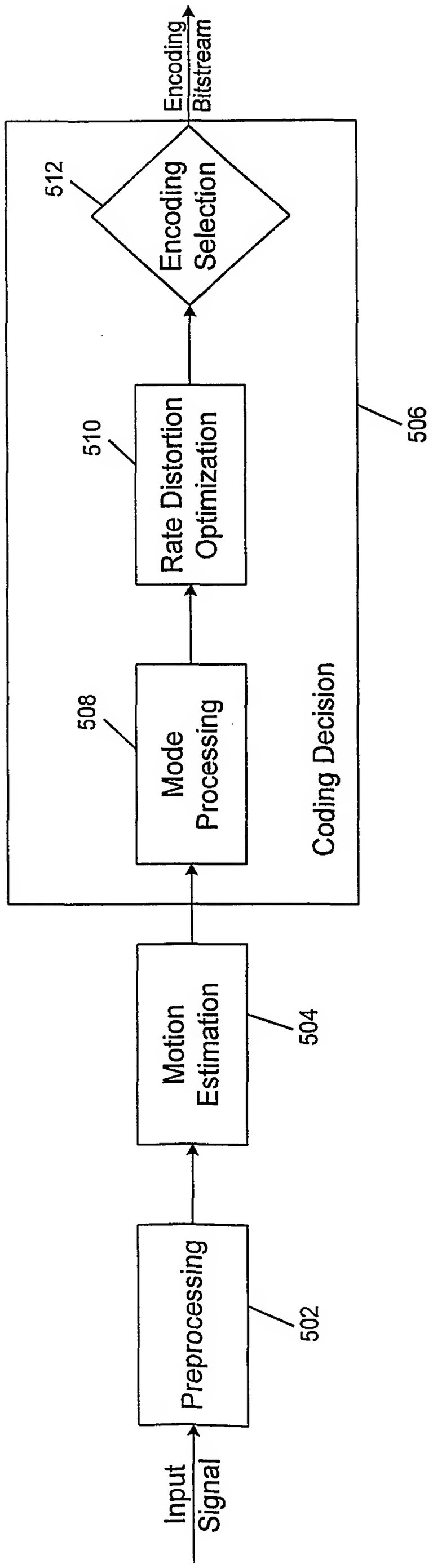

Input Signal
Preprocessing
502
Motion Estimation
504
Coding Decision
506
Mode Processing
508
Rate Distortion Optimization
510
Encoding Selection
512
Encoding Bitstream

FIGURE 5

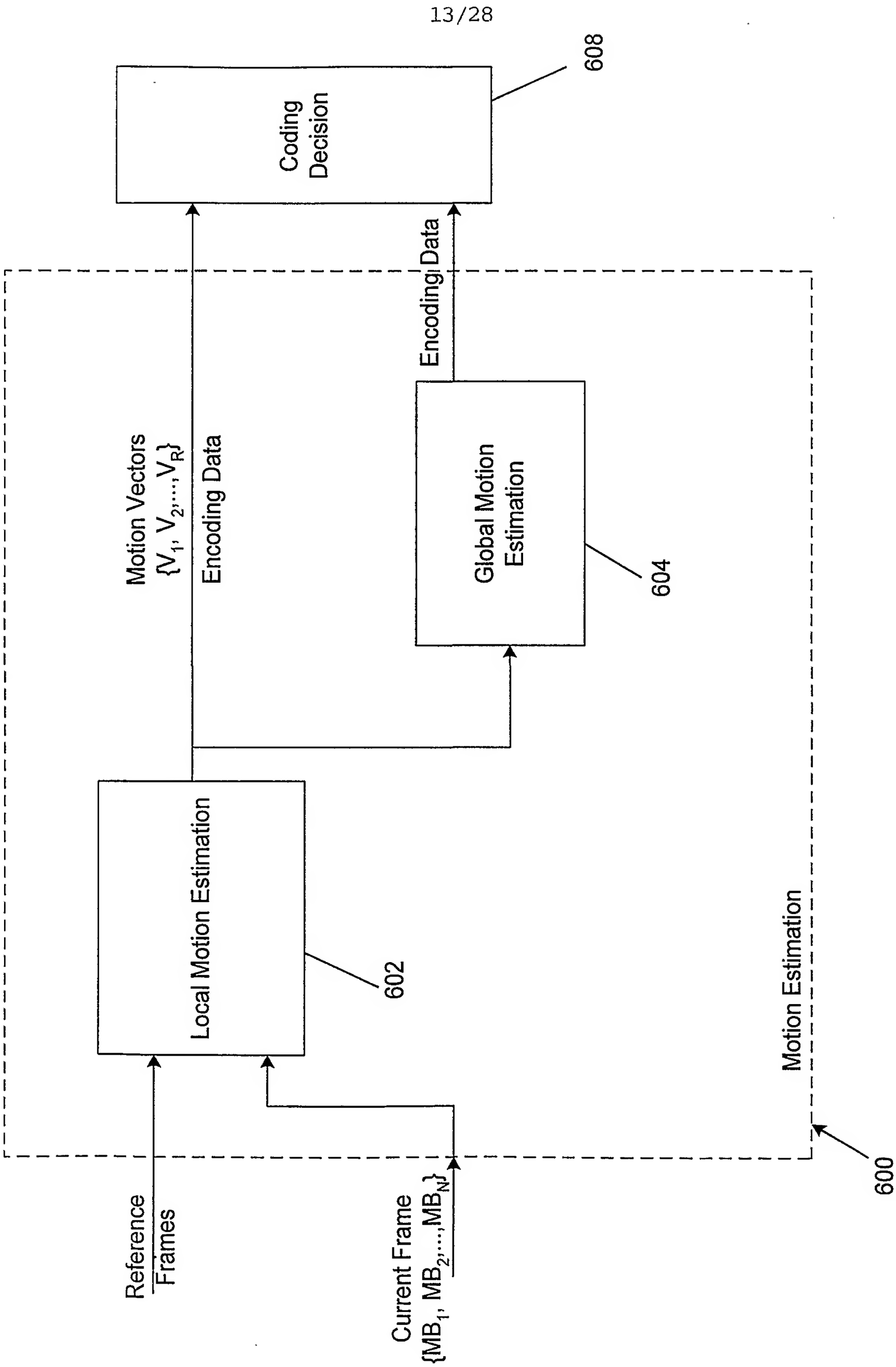

Coding Decision
608
Encoding Data
Motion Vectors
$\{V_1, V_2, \ldots, V_R\}$
Encoding Data
Global Motion Estimation
604
Local Motion Estimation
602
Motion Estimation
600
Reference Frames
Current Frame
$\{MB_1, MB_2, \ldots, MB_N\}$

FIGURE 6A

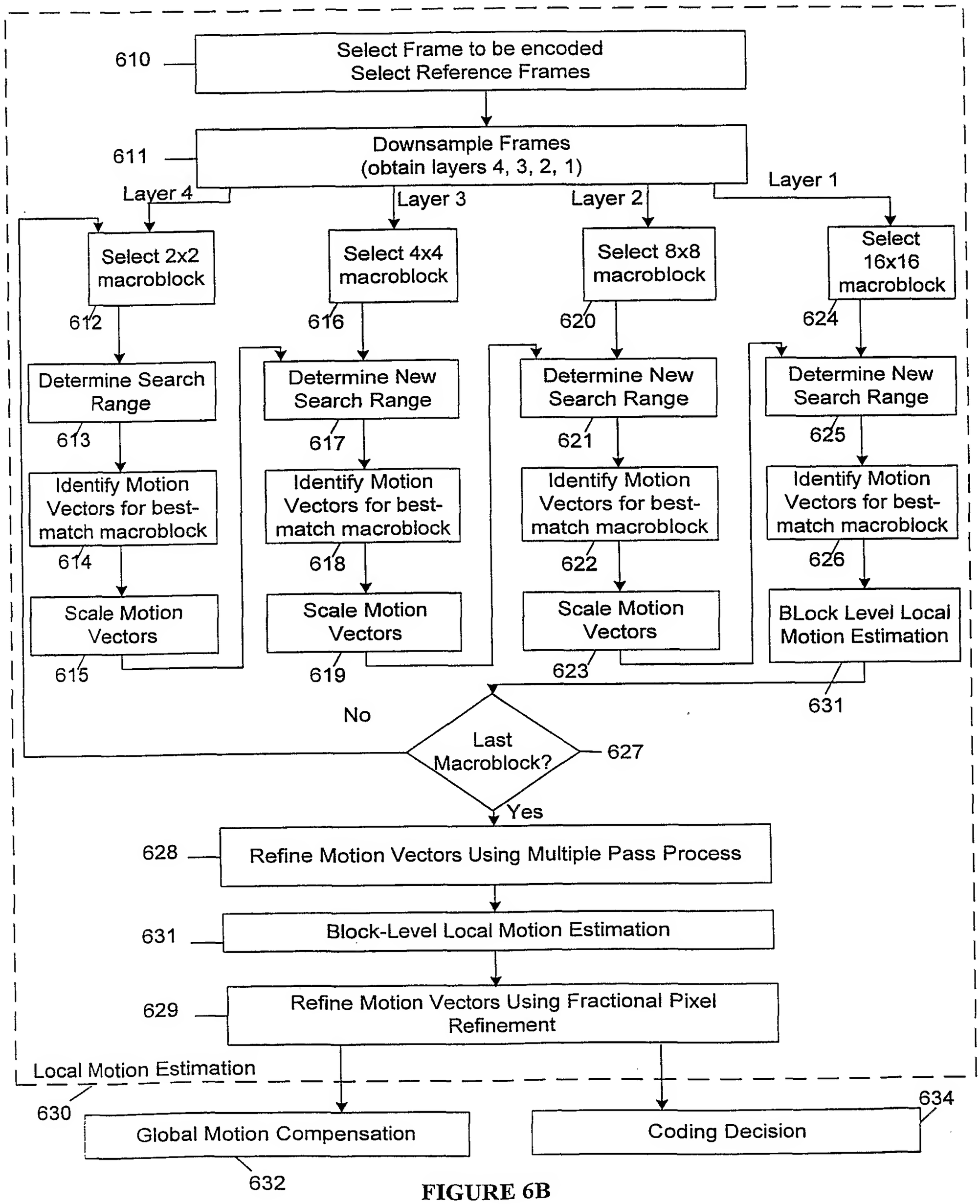

610
Select Frame to be encoded
Select Reference Frames
611
Downsample Frames
(obtain layers 4, 3, 2, 1)
Layer 4
Layer 3
Layer 2
Layer 1
Select 2x2 macroblock
Select 4x4 macroblock
Select 8x8 macroblock
Select 16x16 macroblock
612
616
620
624
Determine Search Range
Determine New Search Range
Determine New Search Range
Determine New Search Range
613
617
621
625
Identify Motion Vectors for best-match macroblock
Identify Motion Vectors for best-match macroblock
Identify Motion Vectors for best-match macroblock
Identify Motion Vectors for best-match macroblock
614
618
622
626
Scale Motion Vectors
Scale Motion Vectors
Scale Motion Vectors
BLock Level Local Motion Estimation
615
619
623
631
No
Last Macroblock?
627
Yes
628
Refine Motion Vectors Using Multiple Pass Process
631
Block-Level Local Motion Estimation
629
Refine Motion Vectors Using Fractional Pixel Refinement
Local Motion Estimation
630
634
Global Motion Compensation
Coding Decision
632

FIGURE 6B

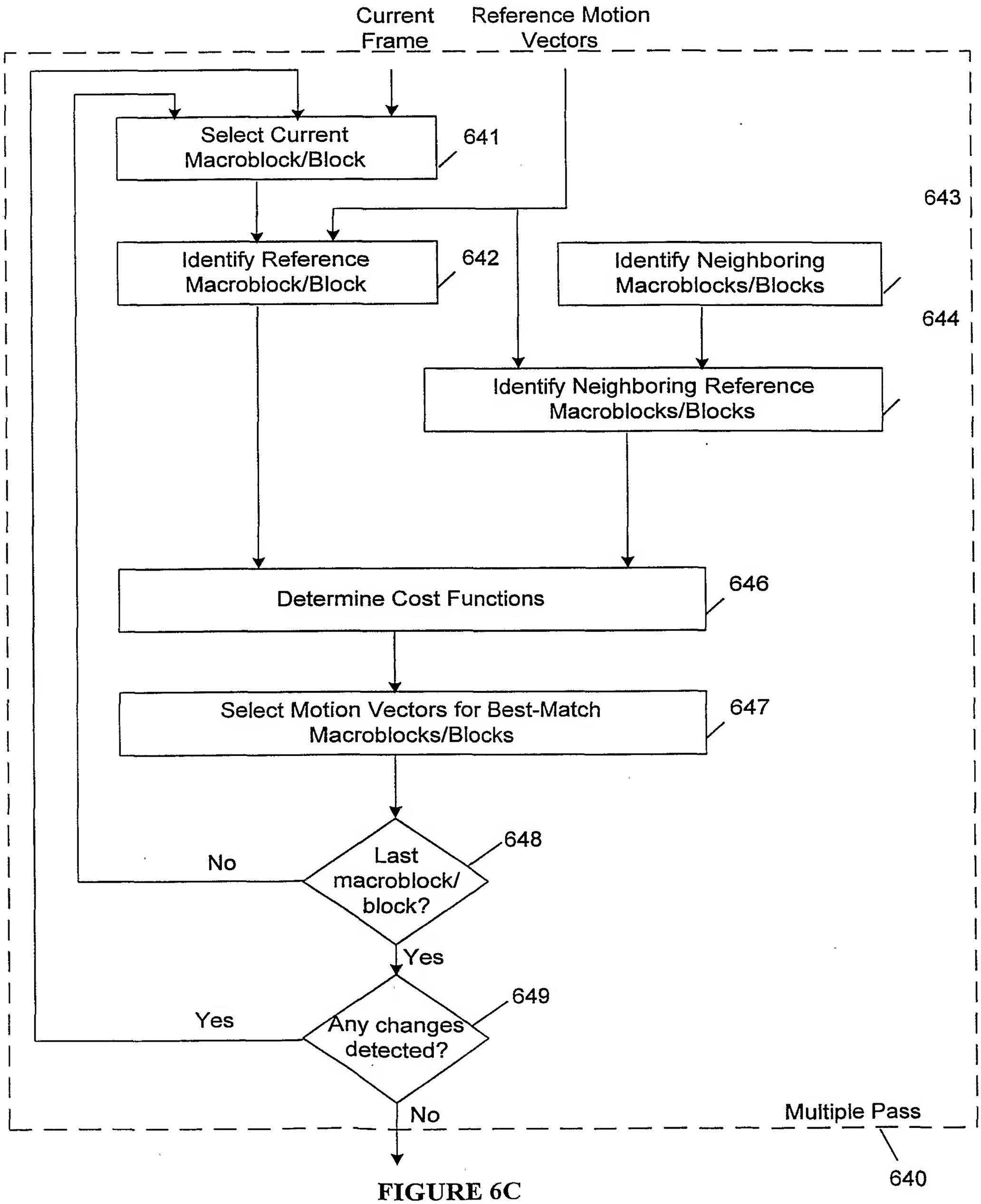

Current Frame
Reference Motion Vectors
Select Current Macroblock/Block
641
Identify Reference Macroblock/Block
642
643
Identify Neighboring Macroblocks/Blocks
644
Identify Neighboring Reference Macroblocks/Blocks
Determine Cost Functions
646
Select Motion Vectors for Best-Match Macroblocks/Blocks
647
Last macroblock/ block?
648
No
Yes
Any changes detected?
649
Yes
No
Multiple Pass
640

FIGURE 6C

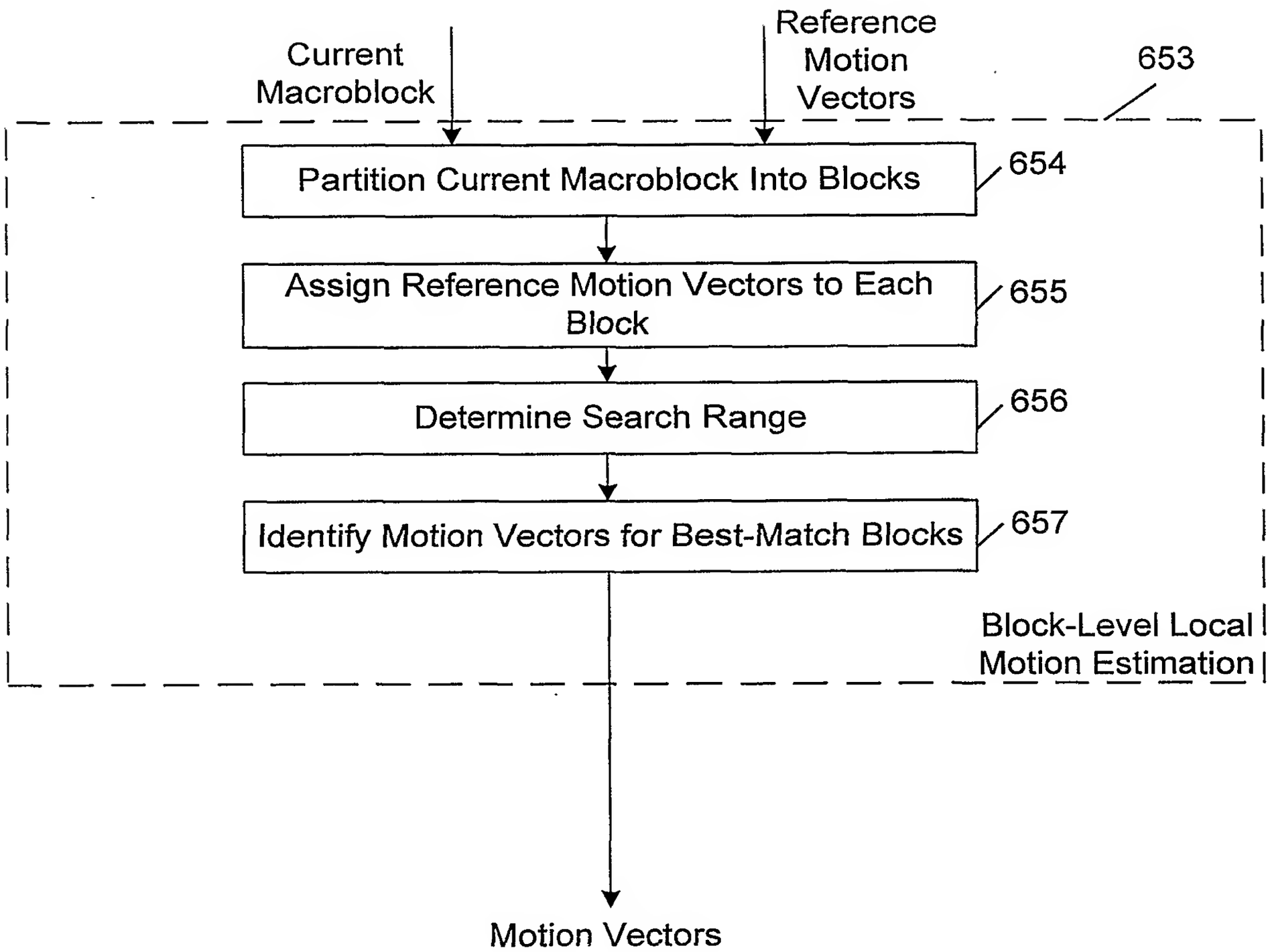

Current Macroblock
Reference Motion Vectors
653
Partition Current Macroblock Into Blocks
654
Assign Reference Motion Vectors to Each Block
655
Determine Search Range
656
Identify Motion Vectors for Best-Match Blocks
657
Block-Level Local Motion Estimation
Motion Vectors

FIGURE 6D

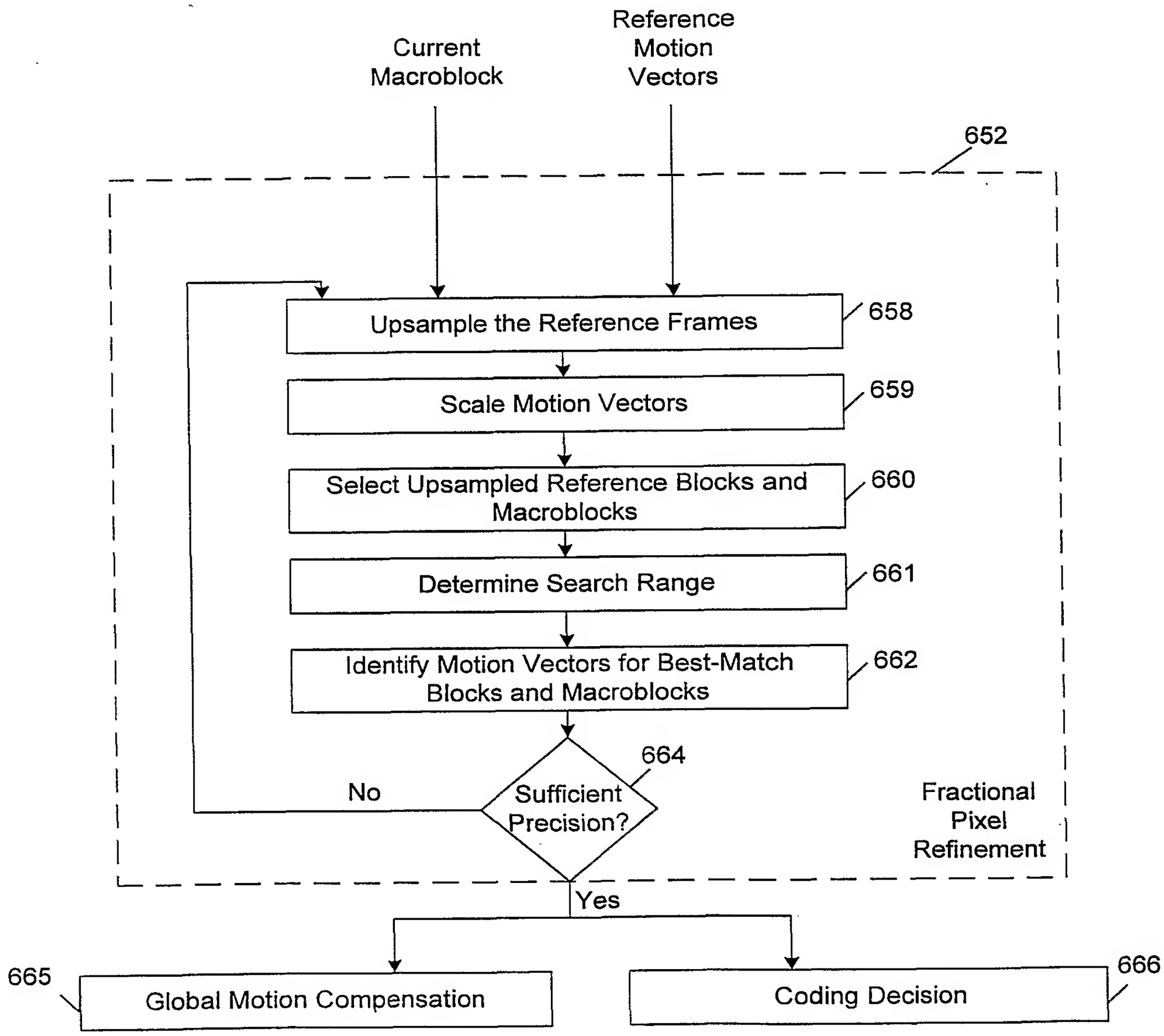

Current Macroblock
Reference Motion Vectors
652
Upsample the Reference Frames
658
Scale Motion Vectors
659
Select Upsampled Reference Blocks and Macroblocks
660
Determine Search Range
661
Identify Motion Vectors for Best-Match Blocks and Macroblocks
662
664
Sufficient Precision?
No
Yes
Fractional Pixel Refinement
665
Global Motion Compensation
666
Coding Decision

FIGURE 6E

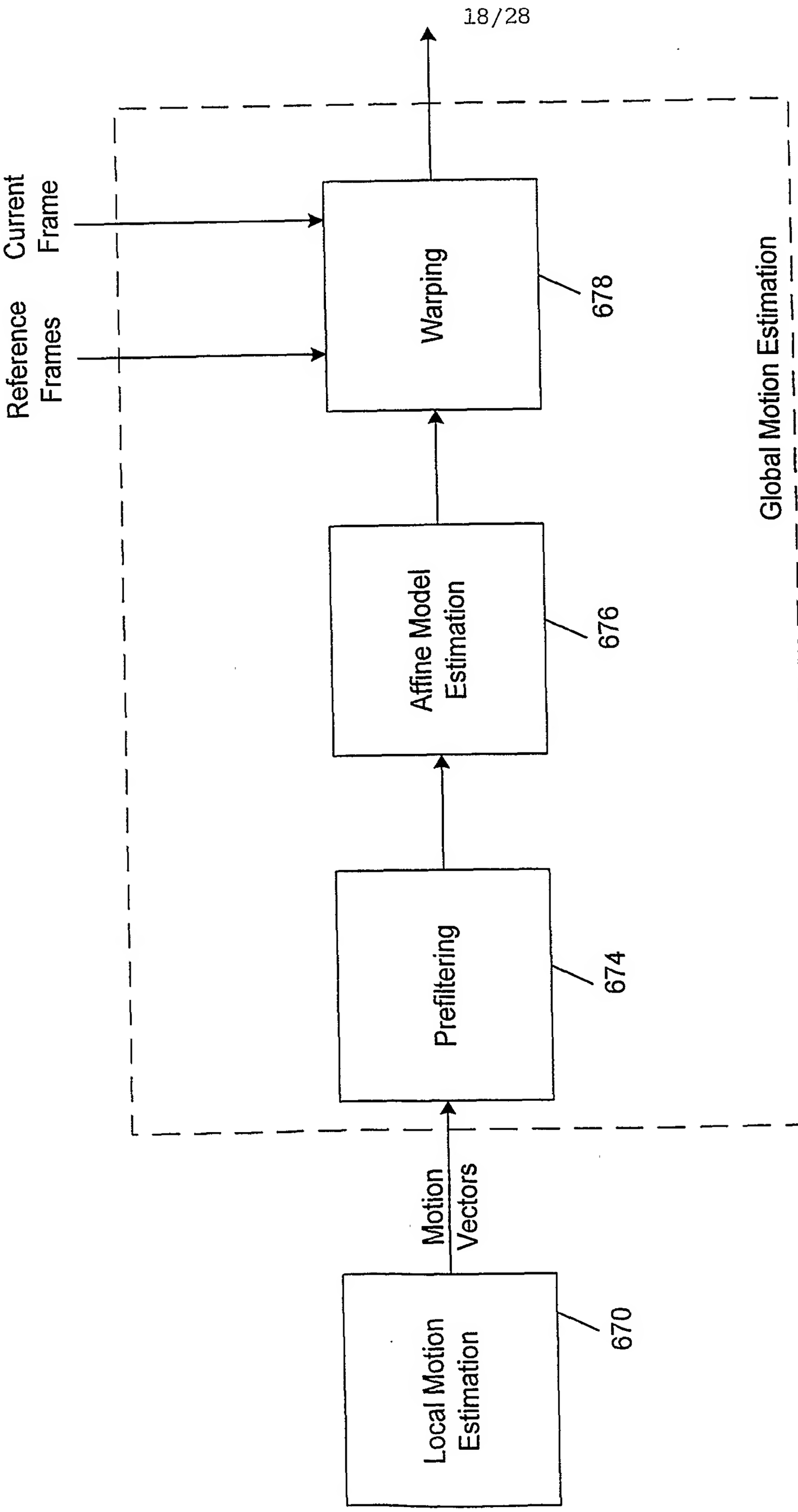

Current Frame
Reference Frames
Warping
678
Affine Model Estimation
676
Prefiltering
674
Global Motion Estimation
672
Motion Vectors
Local Motion Estimation
670

FIGURE 6F

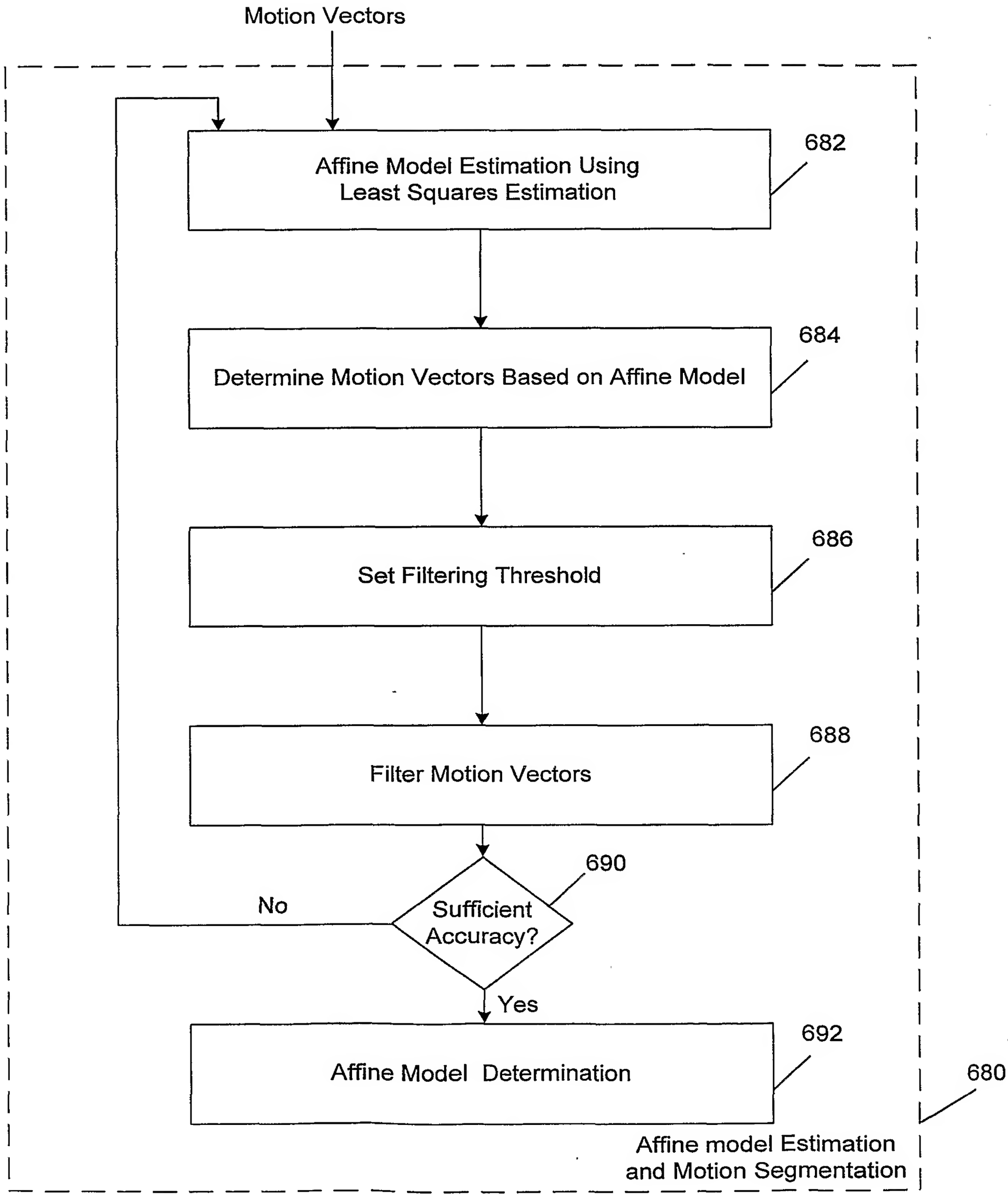

Motion Vectors
Affine Model Estimation Using Least Squares Estimation
682
Determine Motion Vectors Based on Affine Model
684
Set Filtering Threshold
686
Filter Motion Vectors
688
Sufficient Accuracy?
690
No
Yes
Affine Model Determination
692
Affine model Estimation and Motion Segmentation
680

FIGURE 6G

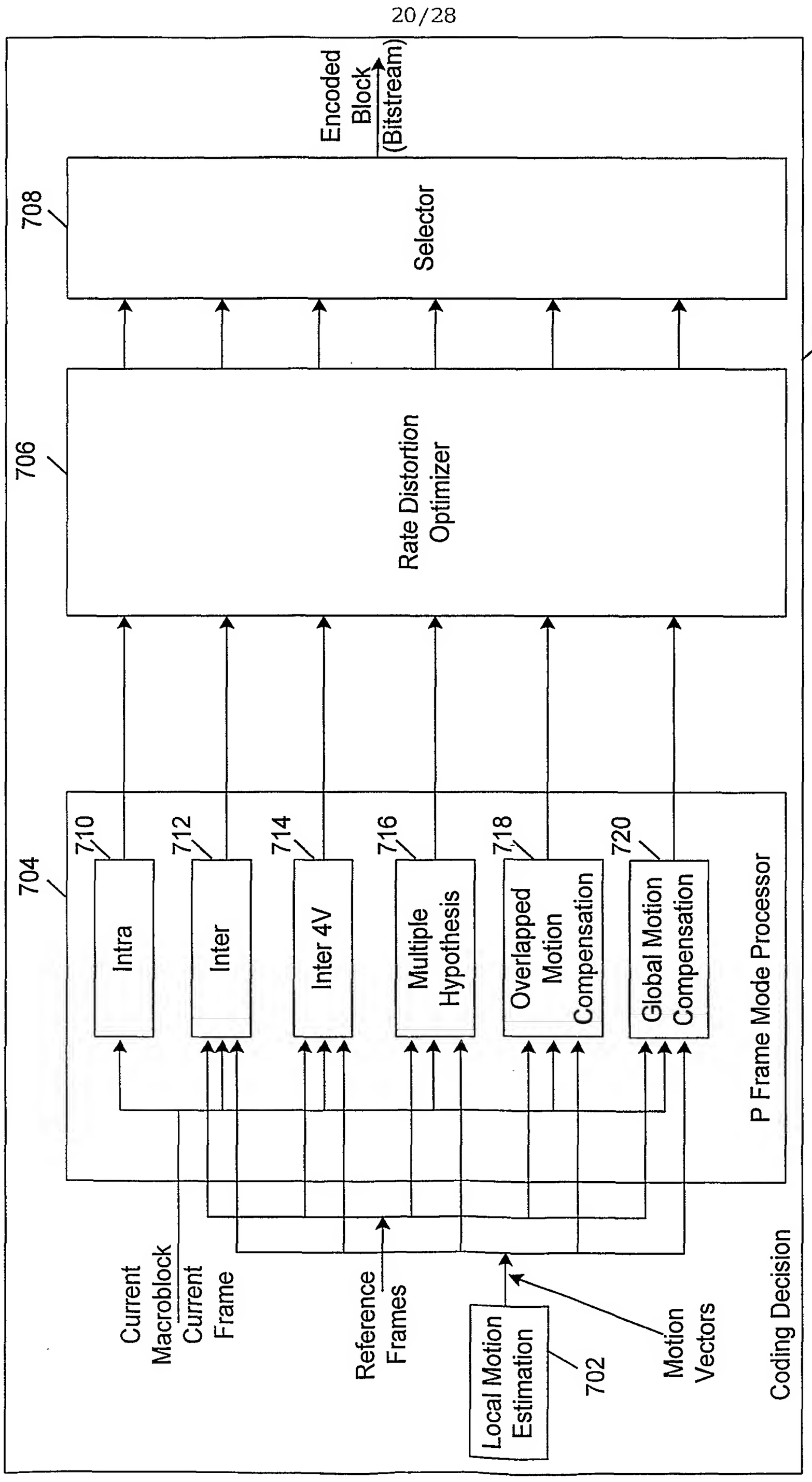

Encoded Block (Bitstream)
708
Selector
700
706
Rate Distortion Optimizer
704
710
Intra
712
Inter
714
Inter 4V
716
Multiple Hypothesis
718
Overlapped Motion Compensation
720
Global Motion Compensation
P Frame Mode Processor
Current Macroblock
Current Frame
Reference Frames
Local Motion Estimation
702
Motion Vectors
Coding Decision

FIGURE 7A

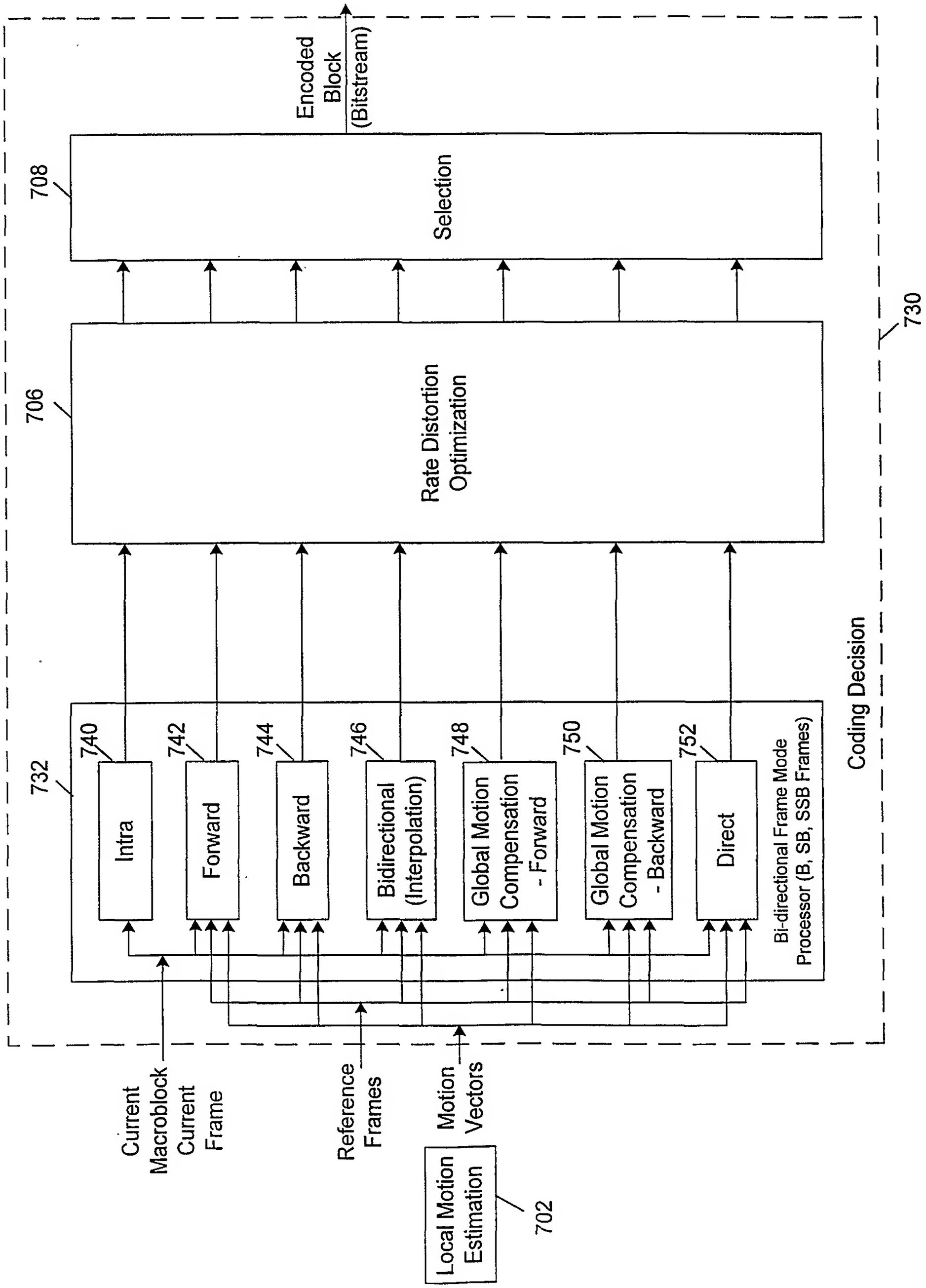

Encoded Block (Bitstream)
708
Selection
730
706
Rate Distortion Optimization
732
740
Intra
742
Forward
744
Backward
746
Bidirectional (Interpolation)
748
Global Motion Compensation - Forward
750
Global Motion Compensation - Backward
752
Direct
Bi-directional Frame Mode Processor (B, SB, SSB Frames)
Coding Decision
Current Macroblock
Current Frame
Reference Frames
Motion Vectors
Local Motion Estimation
702

FIGURE 7B

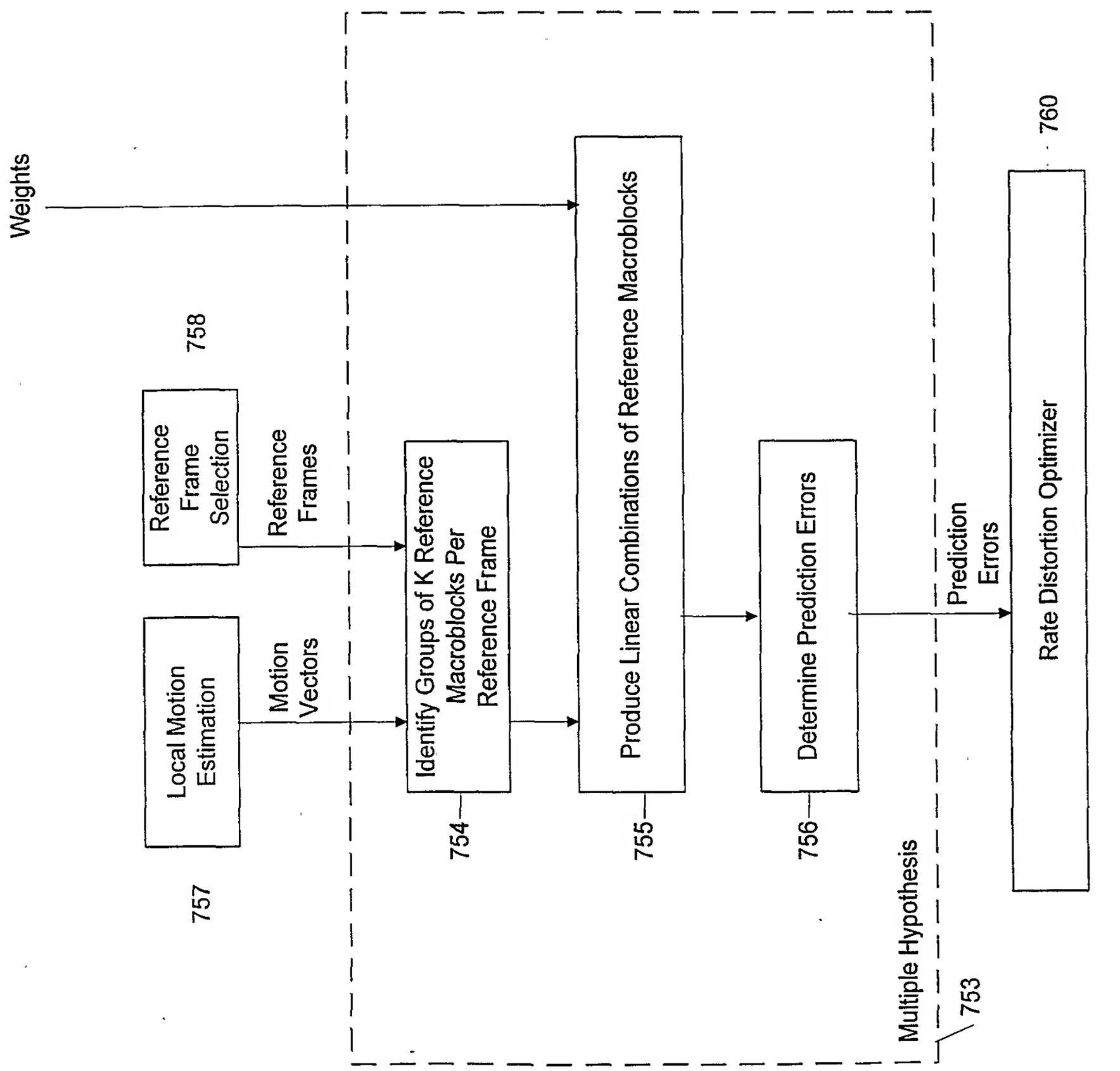

Weights
Reference Frame Selection
758
Reference Frames
Local Motion Estimation
Motion Vectors
Identify Groups of K Reference Macroblocks Per Reference Frame
754
Produce Linear Combinations of Reference Macroblocks
755
Determine Prediction Errors
756
Prediction Errors
Rate Distortion Optimizer
760
757
Multiple Hypothesis
753

FIGURE 7C

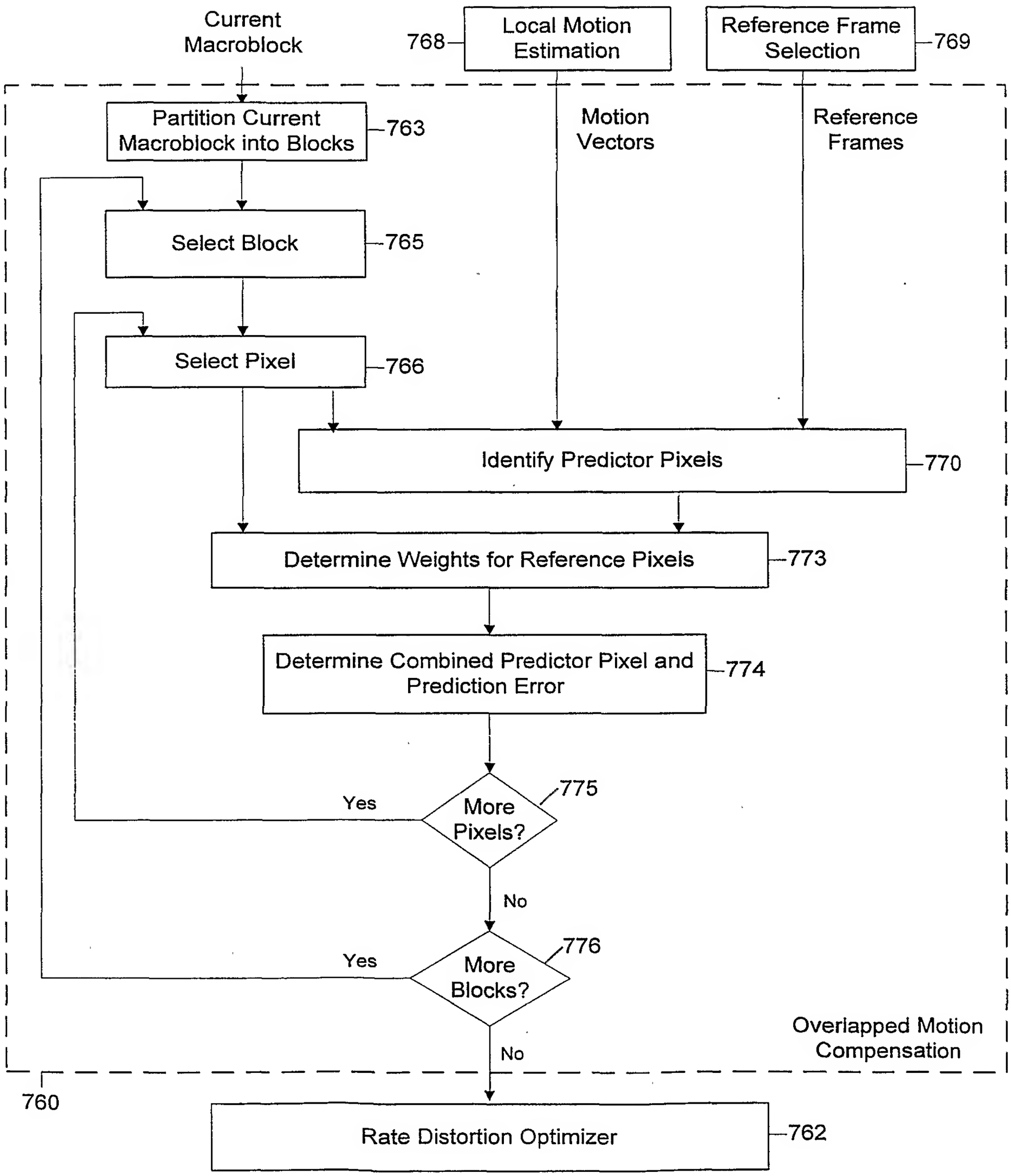

Current Macroblock
768
Local Motion Estimation
Reference Frame Selection
769
Partition Current Macroblock into Blocks
763
Motion Vectors
Reference Frames
Select Block
765
Select Pixel
766
Identify Predictor Pixels
770
Determine Weights for Reference Pixels
773
Determine Combined Predictor Pixel and Prediction Error
774
Yes
More Pixels?
775
No
Yes
More Blocks?
776
No
Overlapped Motion Compensation
760
Rate Distortion Optimizer
762

FIGURE 7D

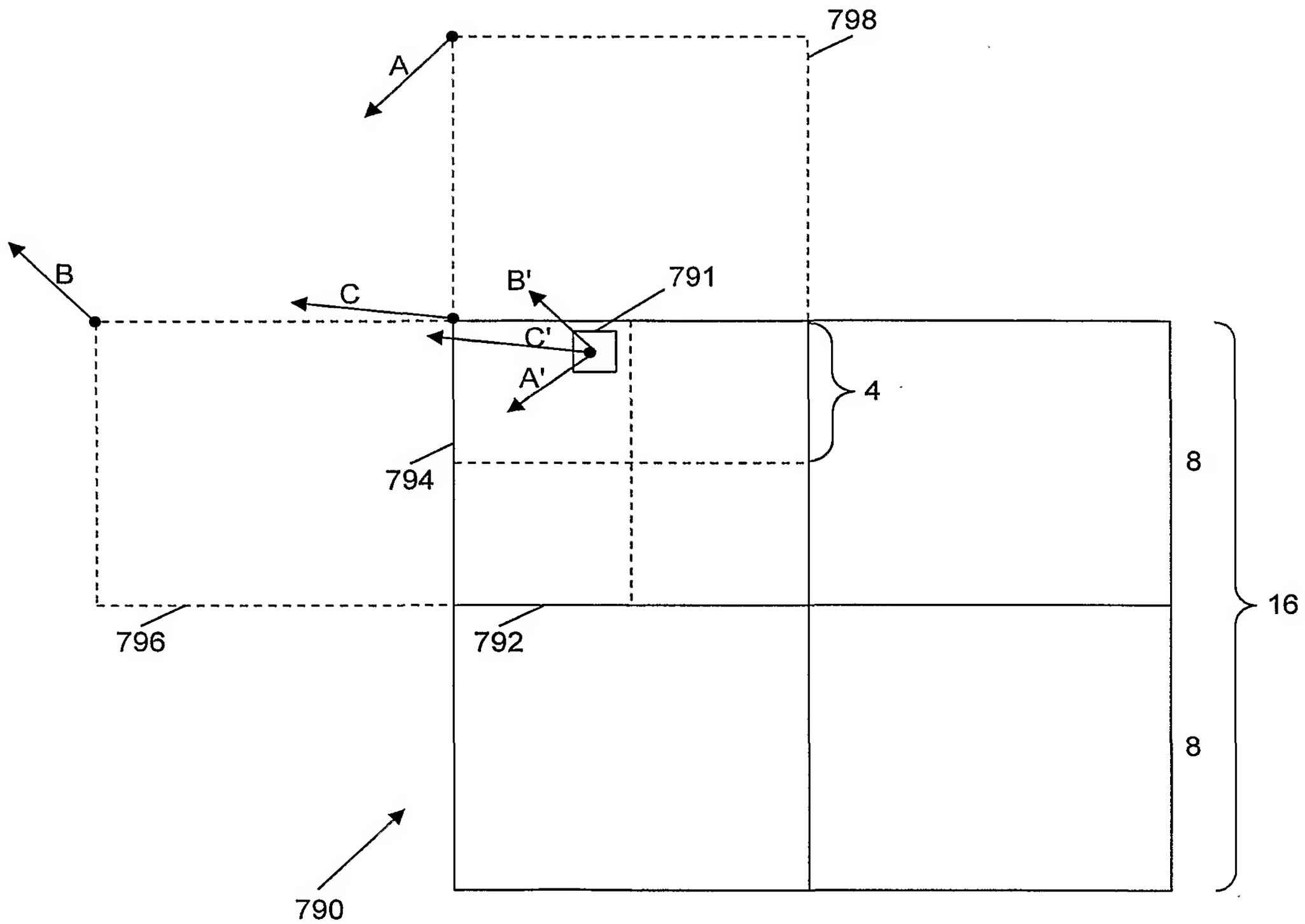


FIGURE 7E

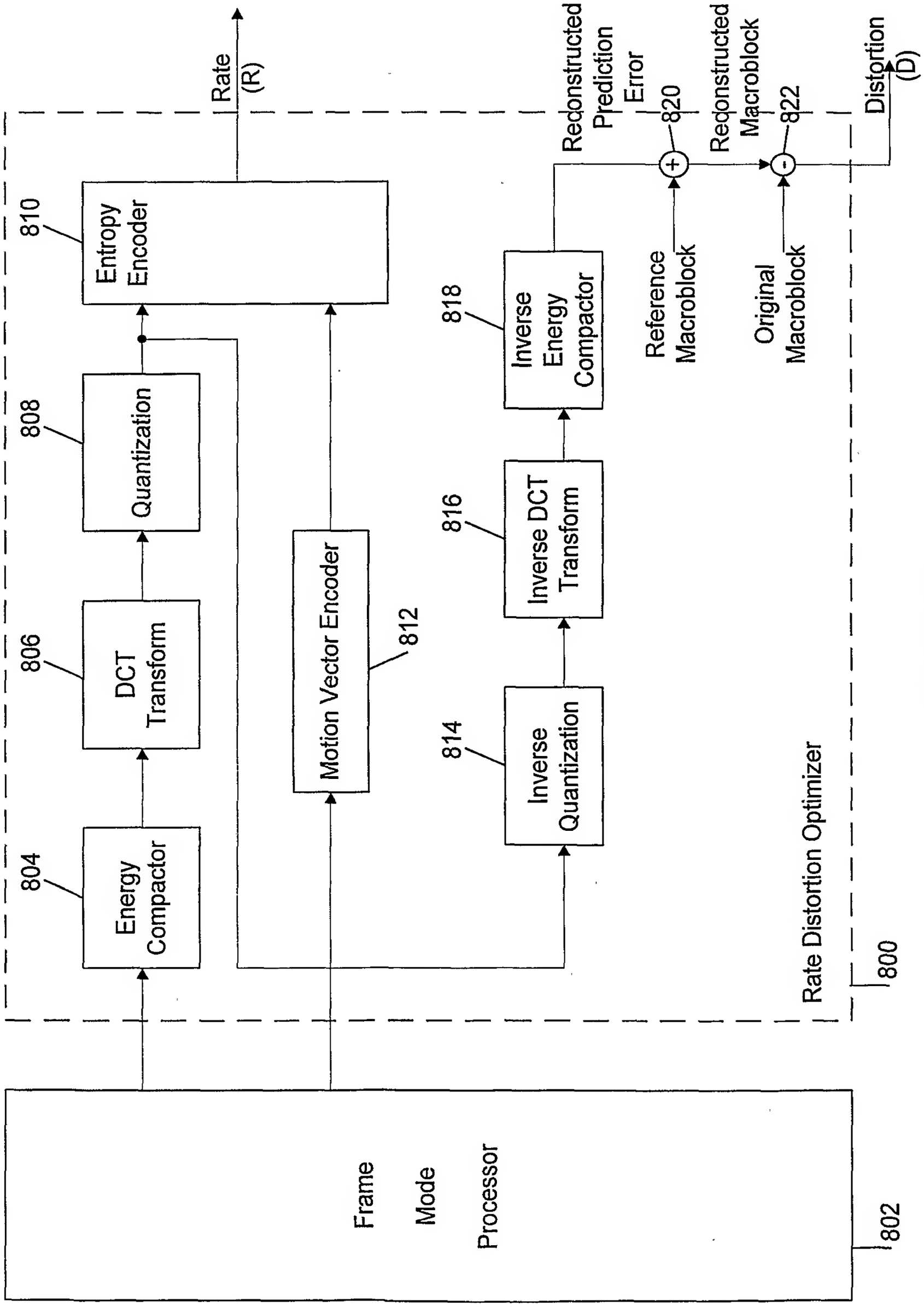

Rate
(R)
Reconstructed
Prediction
Error
820
Reconstructed
Macroblock
822
Distortion
(D)
810
Entropy
Encoder
818
Inverse
Energy
Compactor
Reference
Macroblock
Original
Macroblock
808
Quantization
806
DCT
Transform
Motion Vector Encoder
812
816
Inverse DCT
Transform
814
Inverse
Quantization
804
Energy
Compactor
Rate Distortion Optimizer
800
Frame
Mode
Processor
802

FIGURE 8A

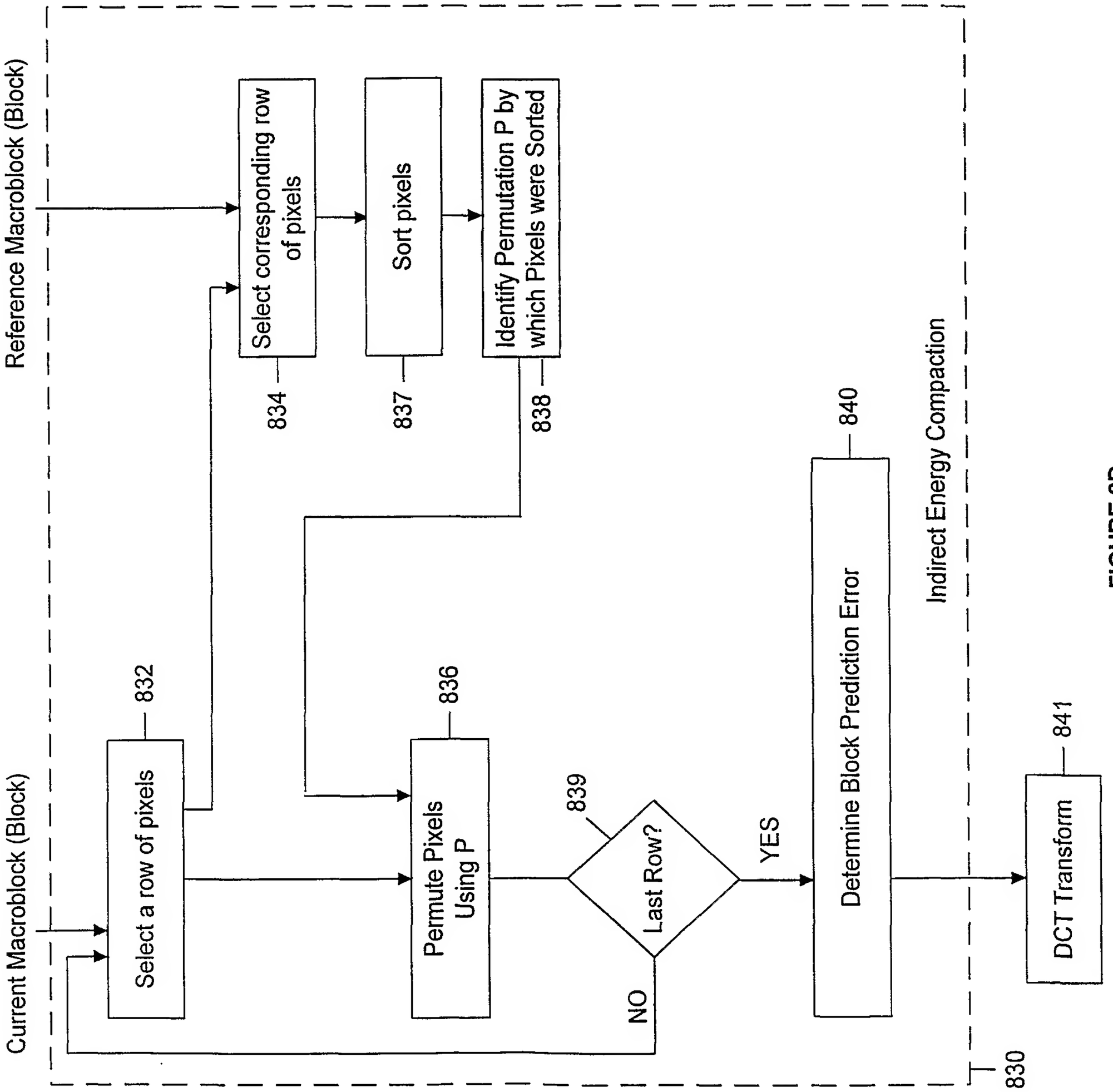

Current Macroblock (Block)
Reference Macroblock (Block)
Select a row of pixels
832
Permute Pixels Using P
836
Last Row?
839
NO
YES
Select corresponding row of pixels
834
Sort pixels
837
Identify Permutation P by which Pixels were Sorted
838
Determine Block Prediction Error
840
DCT Transform
841
Indirect Energy Compaction
830

FIGURE 8B

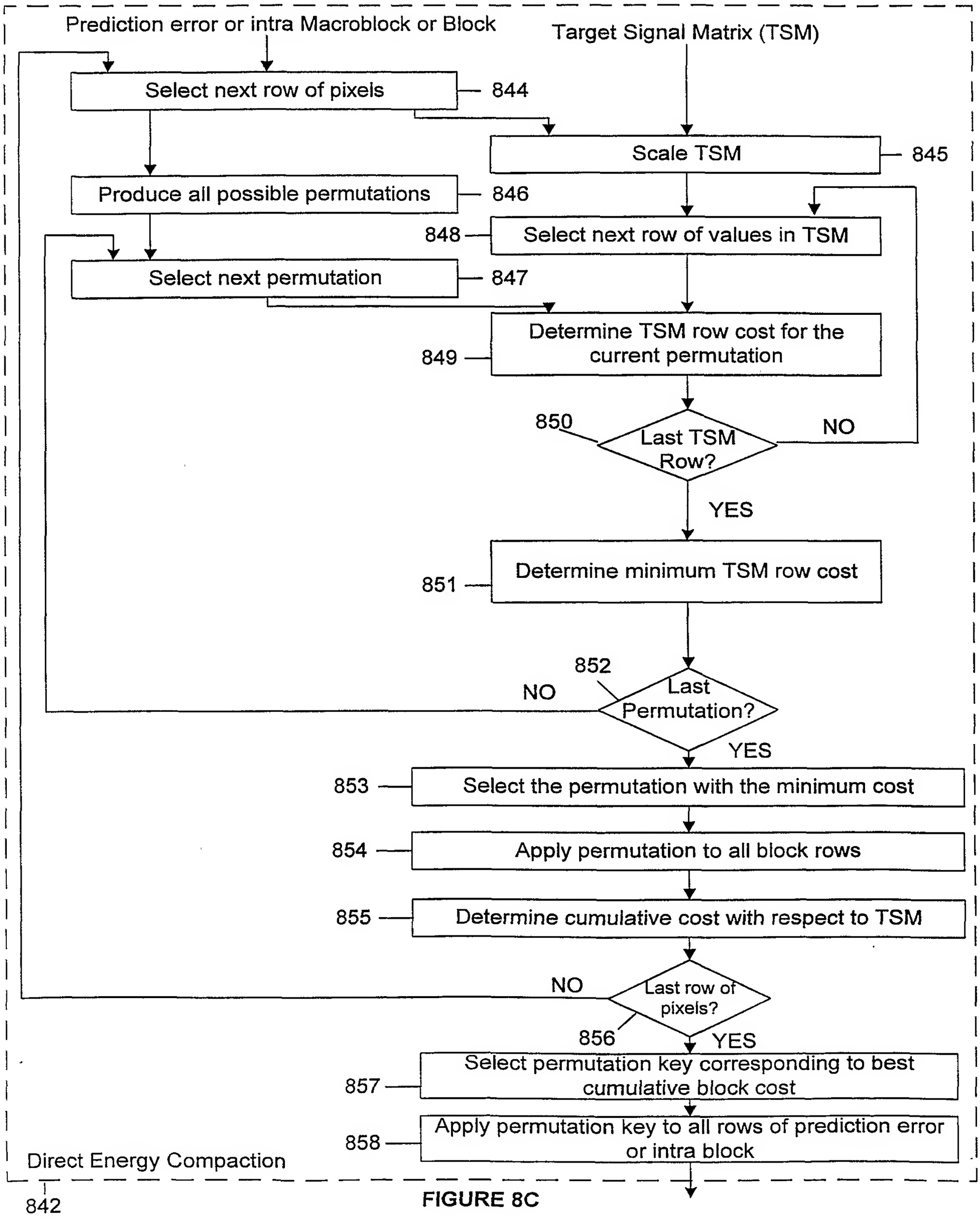

Prediction error or intra Macroblock or Block
Target Signal Matrix (TSM)
Select next row of pixels
844
Scale TSM
845
Produce all possible permutations
846
848
Select next row of values in TSM
Select next permutation
847
Determine TSM row cost for the current permutation
849
850
Last TSM Row?
NO
YES
Determine minimum TSM row cost
851
852
Last Permutation?
NO
YES
853
Select the permutation with the minimum cost
854
Apply permutation to all block rows
855
Determine cumulative cost with respect to TSM
NO
Last row of pixels?
856
YES
857
Select permutation key corresponding to best cumulative block cost
858
Apply permutation key to all rows of prediction error or intra block
Direct Energy Compaction
842

FIGURE 8C

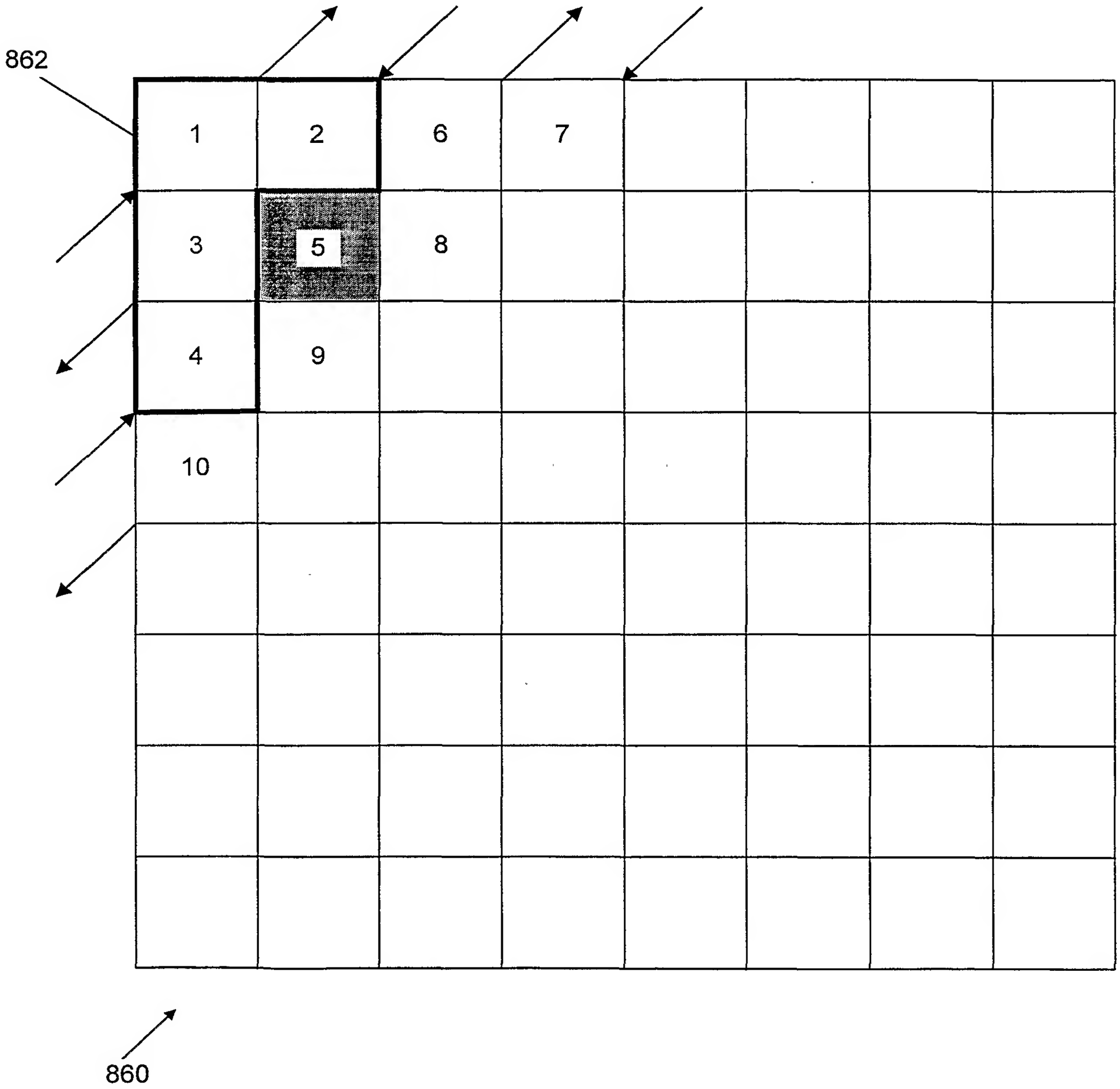


FIGURE 8D